AFRL-IF-RS-TR-2001-264
Final Technical Report
January 2002

ASSOCIATIVE MEMORY STUDY: ARCHITECTURES AND TECHNOLOGY

University of Pittsburgh

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2001-264 has been reviewed and is approved for publication.

APPROVED:

BERNARD J. CLARKE
Project Engineer

FOR THE DIRECTOR:

JOSEPH CAMERA, Chief
Information & Intelligence Exploitation Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFED, 32 Brooks Road, Rome, NY 13441-4114. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	JANUARY 2002	Final Sep 99 - Aug 00

4. TITLE AND SUBTITLE
ASSOCIATIVE MEMORY STUDY: ARCHITECTURES AND TECHNOLOGY

5. FUNDING NUMBERS
C - F30602-99-1-0556
PE - 62702F
PR - PMEM
TA - 00
WU - 03

6. AUTHOR(S)
Steven P. Levitan, Donald M. Chiarulli, Amirtha Katsuri, and Joan Kettering

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Pittsburgh
Office of Grants and Contracts
350 Thackaray Hall
Pittsburgh Pennsylvania 15260

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/IFED
32 Brooks Road
Rome New York 13441-4114

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2001-264

11. SUPPLEMENTARY NOTES
Air Force Research Laboratory Project Engineer: Bernard J. Clarke/IFED/(315) 330-2106

12a. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*
This report summarizes current work in technologies and architectures for associative or "content addressable" memory. Associative memory is better suited for several specific tasks such as database applications than conventional memory. However, its higher cost and complexity has, until now, limited its use to small, special purpose applications such as translation look-aside buffer used in a computer virtual memory system or network router tables. The conclusions indicate the using new optical technology in associative processors is a promising approach.

14. SUBJECT TERMS
Associative Memory, Content Addressable Memory, Content Addressable Parallel Processors, Optical Associative Memory

15. NUMBER OF PAGES
112

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Standard Form 298 (Rev. 2-89) (EG)
Prescribed by ANSI Std. 239.18
Designed using Perform Pro, WHS/DIOR, Oct 94

Table of Contents

1 Executive Summary

This report summarizes current work in technologies and architectures for associative or "content addressable" memory. Associative memory is better suited for several specific tasks such as database applications than conventional memory. However, its higher cost and complexity has until now limited its use to small, special purpose applications such as the translation look-aside buffer used in a computer virtual memory system or network router tables. We conclude that using new optical technology in associative processors is a promising approach.

2 Introduction

This study examines the latest associative memory technologies and architectures. While the concepts behind associative or content addressable memories have been around for over 40 years, they have always been deemed too expensive to justify their wide acceptance. On the other hand, for particular applications these memories have the ability to greatly reduce the processing time. This is for two reasons. First, the memory organization directly supports parallel operations reducing the inherent algorithm complexity by a factor of O(n). Second, the operations are done directly in hardware, providing what is sometimes called "smart memory" operations, thus further reducing the processing time.

One of the most severe drawbacks of associative memories is the problem of size. The advantage of associative memories can only be realized when the data set "fits" entirely into the memory. The uses of either streaming I/O or caching of the data are inherently serial and thus defeat the advantages of the parallel nature of the memory. None the less, as we discuss below, many proposed "associative memories" are really streaming memories connected to specialized search hardware. The real solutions to this problem are either to have a parallel load functionality that can be done at processing speeds, or simply to build large enough memories to fit the entire application.

This study also reviews algorithms and applications using associative memory for tasks of interest to the Air Force, such as database operations and image analysis.

This study is organized as follows. Section 3 explains the general concept of associative memory. Section 4 describes various associative memory organizations and Section 5 discusses associative processors and the kind of programming and algorithms required to effectively use their associative memory. The next two sections discuss various technologies used to implement associative memories - section 6 covers semiconductors and section 7 covers optics. Neural networks, which are specialized associative processing systems, are described in section 8. Section 9 briefly discusses some other applications and section 11 contains the conclusions of this study. Appendix I presents key concepts in relational databases as background material, and Appendix II provides an in-depth discussion of a particular set of Image processing applications on a specialized processor. The report ends with an annotated bibliography.

3 General Concepts

This section begins the study by defining associative memory and discussing its advantages and disadvantages.

In a conventional address-based memory, each word is assigned a numerical address that is used to access the word. For example, reading from address 1000H will copy the contents of the word with address 1000H into a register. Each memory port usually accesses words serially.

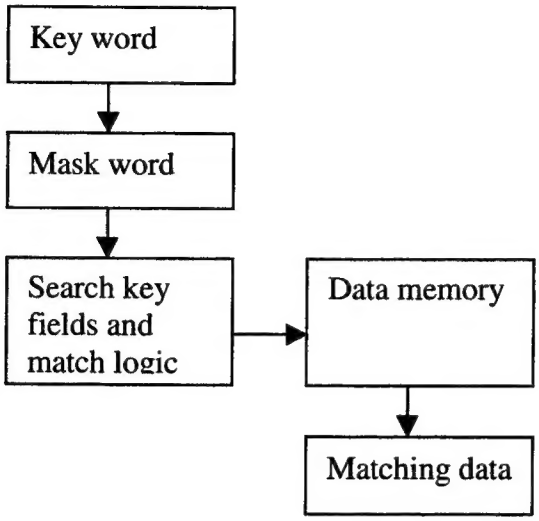
In an associative memory, words are stored and retrieved based on their contents, not on their addresses. Words in associative memory are usually accessed in parallel. For example, all words containing '14' will respond to a request to match '14'. Associative memory is also called context-addressable memory.

Foster[GEN2] provides the following analogy to contrast the two memory organizations. A professor is trying to find out if any of his students have a particular book. In the conventional approach, he asks, "Does the student in seat 1 have this book? Does the student in seat 2 have this book? ..." and so on, one at a time. Using the associative approach, he stands before the class and says, "If you have a copy of this book, please hold up your hand."

In order to meet the above definition, an associative memory must provide at least the following functions:

- Broadcast of a search argument to all locations
- Comparison of the search argument with the contents of all locations
- Identification of the matching words and, if necessary, prioritizing multiple matching words
- Read/write access to the matching words

The following block diagram of a sample associative memory that provides all of these functions is based on Chisvin[GEN1] and Grosspietsh[GEN3].



The key word contains the pattern to be compared with the memory contents. The mask word masks all but the relevant bits of the key word. The actual data comparison takes place between the masked key word and the search key fields in parallel using the matching logic. Data memory corresponding to a match can then be read or written.

Each memory word can be viewed as having two parts – the search key and the data memory. The relative sizes of the two parts can vary. At one extreme, the search key field can be a short tag and the data memory part can be several bytes long. At the other extreme, the entire memory word can be the search key.

The matching logic can look for an exact match or an approximate match that is the "closest" in some way to the key word. Methods for finding an approximate match are described in more detail below. A more complex match function, such as all values within a numerical range, can also be used.

Once matches have been found, the match logic usually provides additional information about the results. Examples of this are indications of no matches, exactly one match, or the number of matches. If there are multiple matches, logic can be used to select one match as the response based on some priority scheme.

To summarize, conventional memory is accessed serially through a single memory access path. This memory access path is called the "von Neumann bottleneck" because it is often the limiting factor for system performance. Associative memory reduces the von Neumann bottleneck in two ways. First, associative memory is searched in parallel rather than serially. Second, some data processing can be performed directly in the associative memory, so that operands do not need to be transferred to and from the memory.

Associative memories are theoretically superior to conventional memories for searching. To search an unordered list stored in conventional memory, the content of each memory location is checked sequentially until a match is found or the end of the list is reached. The complexity of

the calculation is O (n/2) on average. To search an unordered list in a fully associative memory, each search key is compared with the key word in parallel. The complexity of this calculation is O(1).

Semiconductor associative memory requires more hardware than the same amount of conventional memory. This means that associative memory has a higher cost/bit, higher power consumption, and lower storage density. The matching hardware slows down the access time, so the time to read or write a single word of data is longer than with a conventional memory. In addition, the number of interconnections and the length of the traces required to broadcast the key word to all words in the memory limits the size of the memory.

Another disadvantage of associative memory is that it must be loaded serially in the same manner as conventional memory. If all of the information to be searched cannot be loaded in the memory at one time, there is a long delay to page-in the new data from disk or tape. Therefore, unless the data can be loaded into the memory at one time and then used repeatedly, there will be little performance improvement over conventional memory.

Chisvin [GEN1] notes what appears to be the single most significant disadvantage of associative memory, which is the lack of appropriate software. Running existing code on a processor with an associative memory will probably not yield performance improvement. Several methods, including entirely new algorithms, have been proposed to resolve this problem and are discussed later in this paper.

4 Associative Memories

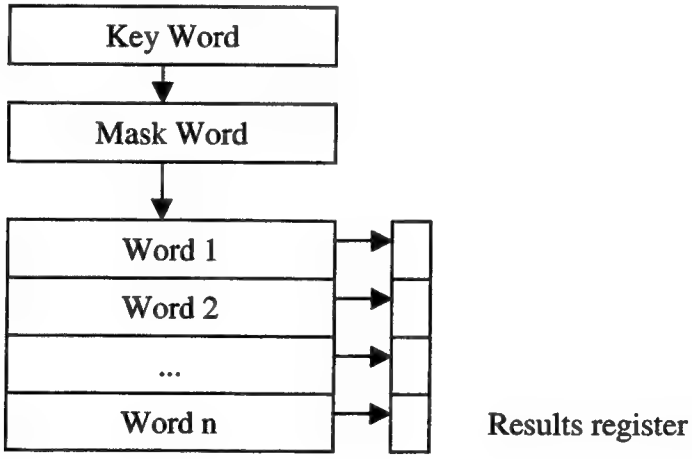
The previous section presented a very general functional block diagram of associative memory. This section becomes more specific and presents details about different memory architectures, matching logic, and dimensions.

Different conventional memory technologies and architectures are used in different applications. Each makes a different price-performance tradeoff. For example, expensive high-speed semiconductor memory is used for cache and lower speed memory is used for main memory. Slower, less expensive sequentially accessed memory such as disk or tape is used for mass storage.

Just as with conventional memory, various associative memory technologies and architectures are available. Architectures that make different price-performance tradeoffs are discussed below and technologies are discussed in section 6.

4.1 Fully Associative

The highest performance architecture is full associativity, where the comparison logic is included in each memory cell. The masked key word is compared simultaneously to all bits of all key fields. If a match is detected in a word, the bit corresponding to that word in the results register is set. Search time in this architecture is O(1).



4.2 Less than Fully Associative

Less than fully associative architectures require less hardware but more time to find a match than a fully associative architecture. The following architectures are all described in Chisvin [GEN1].

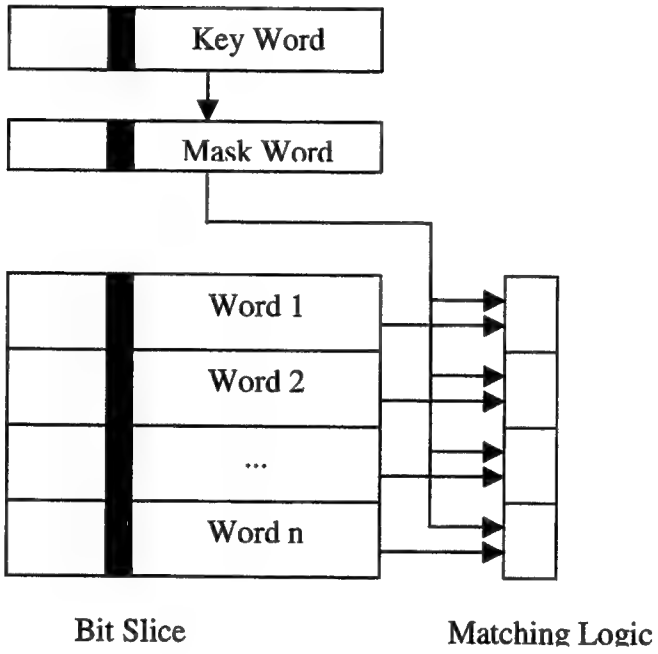
4.2.1 Bit-Serial Architecture

In a bit-serial architecture, a bit slice from each memory key field is inspected simultaneously. For example, bit 0 of all key fields is inspected, bit 1 is inspected, bit 2 is inspected, and so on

until the entire width of the key field is checked. The search time in this architecture is O(w), where w is the key field width in bits.

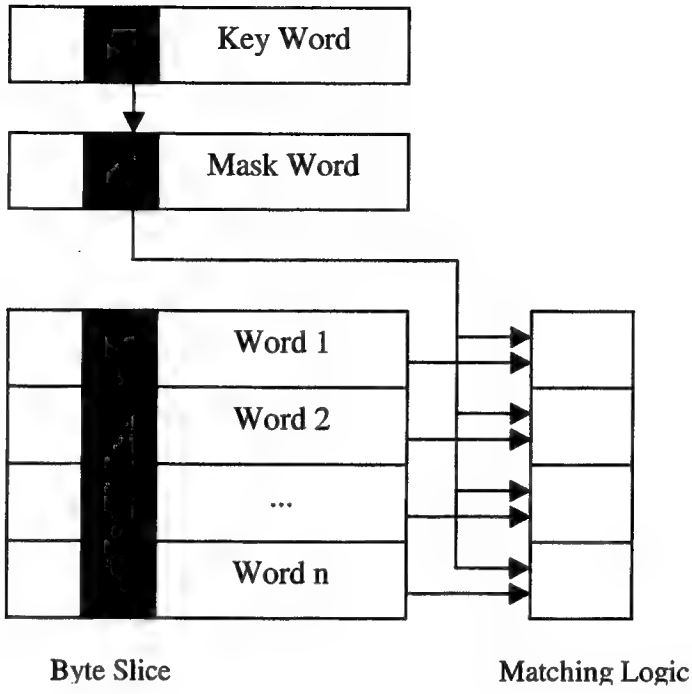
Bit-Serial architectures are common because they can be constructed using off-the-shelf conventional RAM chips.

The block diagram of a bit-serial architecture is shown below. Each word contains both the key field and the associated data memory. The comparison logic is at the output of the memory, not in each word. One bit slice at a time is loaded into the comparison logic and is compared with the appropriate bit from the masked key word.



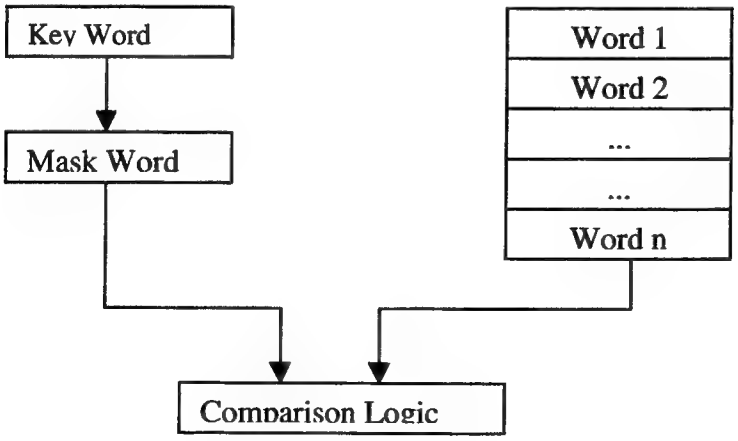
4.2.2 Byte-Serial Architecture

Byte-serial architecture is similar to bit-serial, except that a byte slice from each key field is inspected simultaneously. For example, byte 0 of each key field is inspected, byte 1 is inspected, and so on until the entire key field is checked. The search time in this architecture is O(b), where b is the number of bytes in the key field.



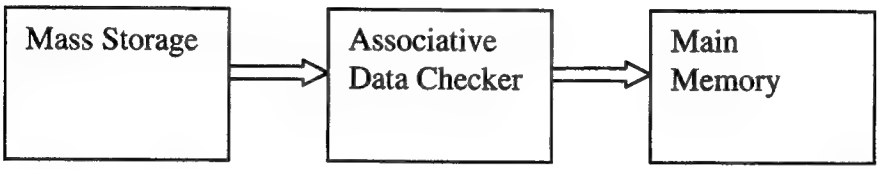
4.2.3 Word-Serial Architecture

A word-serial architecture compares one entire key field to the key word simultaneously, then checks the next key field, and so on until the end of the memory is reached. One memory access therefore cycles through all key fields in the memory sequentially. The search time in this architecture is O(n), where n is the number of words in the memory. This architecture still performs better than a conventional architecture, however, because the comparisons are all done in memory and do not require loading data into the processor registers.



4.2.4 Mass storage systems

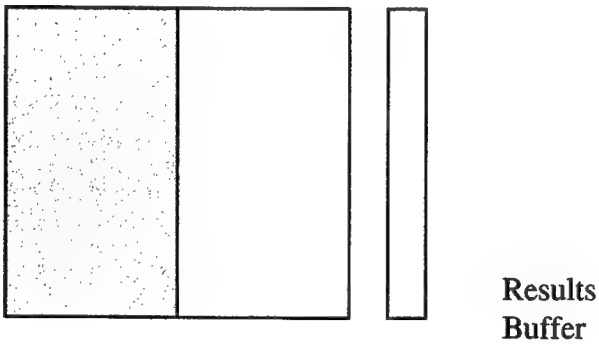
Associative access to data from mass storage systems such as disks or tape can be done by adding hardware at the interface between main memory and the mass storage. The extra hardware checks for matches on the fly as the data is transferred from mass storage. Most proposals for this type of access check a large amount of data, such as a sector from a disk, in parallel and are thus quite fast. Usually, only data blocks containing the desired pattern are written into main memory for further processing.



4.2.5 Adding an Input/Output Buffer

As previously mentioned, a disadvantage of associative memory is that, even though it is searched in parallel, it must be loaded serially. There may be a long delay to reload the memory with new data.

Shain[IMAGE2] proposes, as a solution to this problem, using part of the associative memory as an input/output buffer. This reduces the load and store time, but requires additional memory cells.



The input/output buffer can be loaded or stored serially while other processing is occurring in the remainder of the memory. Once the I/O buffer is ready, it can be transferred to or from the processing array section of the memory one bit-slice at a time, using the results buffer.

4.2.6 One-Dimensional and Two-Dimensional Memory

All of the above examples are one-dimensional associative memories, in which the memory is searched for a single key word. Using optical technology as discussed below, it is possible to build two-dimensional memories. In a two-dimensional associative memory, the memory is searched for several key words at one time. These are discussed in the section on optical technologies.

4.3 Matching Logic

As shown in the basic block diagram in section 3, above, all associative memories contain matching logic. The matching logic will determine whether or not there is an exact or approximate match with the masked search key. The matching logic can also be more complex, such as recognizing magnitude relationships such as less than or between limits.

The mask register can be used to mask out all but the bits of interest, so that the matching occurs on a field rather than the whole search key word.

Exact matching and two methods to find approximate matches are described below.

4.3.1 Exact Matching

Exact matching is a bit-wise negated exclusive or function, as shown in the truth table below.

Masked key word bit	Search key word bit	Result (1 = match)
0	0	1
0	1	0
1	0	0
1	1	1

4.3.2 Weighed Sums for Approximate Matches

Sima [GEN8] provides the following general expression for non-exact matching logic that determines how closely the key word matches the search key:

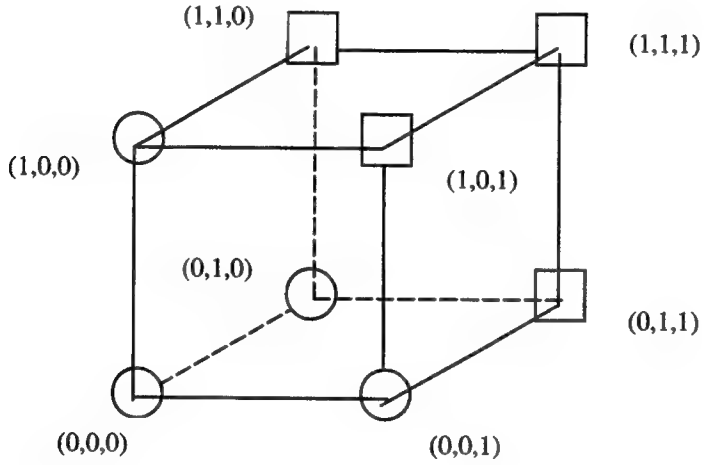
$$F = \Sigma W_i M_i - \Sigma W_j M_j$$

F is the figure of merit for the match. The higher the figure of merit, the more exact the match. Each M_i is a bit that matches and each M_j is a bit that does not match. W_i and W_j are the weights of the matches and mismatches because some bits may have more significance than others.

4.3.3 Hamming Distances for Approximate Matches

Another approach to non-exact matching logic uses Hamming distances, which is based on the well-known Hamming error detection and correction code. Hamming encoding requires appending extra check bits to the data bits. For example, 7 check bits are required to detect and correct one error bit in 32 bits of data.

Hamming encoding can be viewed geometrically. If all possible combinations of data and check bits are enumerated, only a subset is valid. Each invalid combination is geometrically closer to one correct data pattern than any other, as shown in the following example from Chisvin [GEN1].



Assume that the two correct data patterns are (0, 0, 0) and (1, 1, 1). The other corners of the cube are then possible patterns that contain a single bit error. The circled corners are geometrically closer (i.e., have a shorter Hamming distance) to (0, 0, 0), so any of those patterns will be corrected to (0, 0, 0). Similarly, the corners with squares have the shortest Hamming distance to (1, 1, 1) and will be corrected to (1, 1, 1).

The concept of Hamming distances can also be used for inexact matching within an associative memory. The memory can respond with the closest data pattern to a key word. One advantage of this approach is that the memory will always provide one and only one response. A second advantage is that by choosing an appropriate Hamming distance between legitimate data words, a robust associative memory can be constructed.

5 Associative Processors and Algorithms

The previous sections of this paper discuss associative memories as stand-alone components. However, a complete processor is required in order to use the memory. Processors containing associative memory are called associative processors. This section discusses the characteristics of associative processors and algorithms that run on them.

Of course, the one feature that is standard in all associative processors is associative memory. For these processors, associative memory is typically used for data memory, while standard memory in a host processor is typically used to store program instructions.

Another feature is a more sophisticated processing unit associated with each word of the associative memory. In [FOSTER] there is description of a hierarchy of ever more complex processing capability for each word, starting with a simple "match bit" and building to a complete ALU for each word. Having more power at each word accelerates arithmetic and logical operations.

A third common feature is the interconnection of associative memories into partitionable networks. Just as with SIMD (single instruction multiple data) arrays, this permits the partitioning of the data into subsets so that processing can be performed in parallel. Irakliotis[GEN4] provides the following example to demonstrate this concept. In a standard associative processor, a query that requests a count of red cars for each state would be done serially - first find the red cars in Alabama, then find the red cars in Alaska, and so on. With a partitionable network, the central control can direct the array to partition itself by state and then has every partition count red cars in parallel, generating unique sums for each state.

As previously mentioned, custom software is required to take advantage of the performance potential of associative memory because standard algorithms assume conventional memory.

Four methods have been proposed to provide this software. The first method is to use compiler optimization of existing code to better use the associative memory. The second method, which is actually used in the ASP associative processor discussed later in this paper [GEN5], is to provide standard function libraries that are designed to best use the associative memory. Both of these methods permit the use of standard programming approaches but do not fully exploit associative memory. The third method is to use an existing parallel programming language to take advantage of the parallelism inherent in associative memory. The fourth proposed method is to use completely new algorithms and programming approaches specifically designed for associative memory. These last two methods require programmers to learn new programming methods, but present the potential for significant performance improvements.

The rest of this section will provide some example algorithms for use with associative memory. The algorithms are usually significantly different than the standard algorithms to solve the same problems. However, as pointed out by Potter [GEN7], programming using associative memory is, in some senses, easier than programming with conventional memory because pointers and sorting are never required. The algorithms are divided into three sections: mathematical,

database, and symbolic processing algorithms, to show the wide range of computing possible on associative processors.

5.1 Mathematical algorithms

Addition and multiplication algorithms are of interest because associative memories support arithmetic operations directly in the memory, without transferring operands to a processor. However, arithmetic is done in a bit-wise fashion and tends to slow down computationally-intensive programs. For this reason, many proposed associative processors provide a small ALU with each memory word as discussed above.

Algorithms to find the largest number in a list and to maintain a stack are also provided as example mathematical algorithms.

5.1.1 Addition

Addition is a fundamental operation of all computers. The specific problem considered below is the addition of a constant to n words in memory. Assume each number is an unsigned integer coded as b bits.

The following standard algorithm is used with conventional memory and has a complexity of O(n). This algorithm also extensively uses the memory data bus because each addition requires at least one load into the CPU and one store from the CPU.

```
for (each word to be modified)
    word = word + constant
```

The algorithm below, based on Foster[GEN2], is optimized for associative memory and has a complexity of O(b). The algorithm works on a bit-slice at a time – bit 0 in all the words, bit 1 in all the words, bit 2 in all the words, and so on. Also, the algorithm assumes there is a carry bit associated with each word in memory. This algorithm is performed entirely in the associative memory and does not require the CPU.

```
for(each bit)
    if (bit i in constant is 1)
            select memory words with bit i = 1 and carry = 0
        bit i = 0 and carry = 1
            select memory words with bit i = 0 and carry = 0
        bit i = 1
    else
            select memory words with bit i = 0 and carry = 1
        bit i = 1 and carry = 0
            select memory words with bit i = 1 and carry = 1
        bit i = 0
```

5.1.2 Multiplication

The specific multiplication problem considered below is the multiplication of a constant with n words in memory. Assume each number is an unsigned integer coded as b bits.
The following standard algorithm is used with conventional memory and has a complexity of O(n). This algorithm also extensively uses the memory data bus because each addition requires at least one load into the CPU and one store from the CPU.

```
for (each word to be modified)
    word = word * constant
```

The algorithm below, based on Foster[GEN2] and the addition algorithm above, is optimized for associative memory and has a complexity of $O(b^2)$. The algorithm is based on the grade-school shift-and-add multiplication method. The algorithm works on a bit-slice at a time – bit 0 in all the words, bit 1 in all the words, bit 2 in all the words, and so on. . This algorithm, like the previous addition algorithm, is performed entirely in the associative memory and does not require the CPU.

Each word in memory is partitioned into two fields - the multiplier field and the partial product field. The algorithm also assumes that there is a carry bit associated with each word in memory.

```
j = 0
for(each bit)
    select words where bit i is 1
    Add multiplicand to bits n+j thru j of partial product of
          each selected word
    j = j + 1
```

5.1.3 Find Max

A well-known problem in computer science is finding the largest number in an unordered list of n numbers. Assume each number is an unsigned integer coded as b bits.

The following standard algorithm is used with conventional memory and has a complexity of O(n).

```
max = 0
    for (each item in the list)
        if (item > max)
            max = item
```

The following algorithm, also from Foster[GEN2], is optimized for associative memory and has a complexity of O(log b).

```
cutoff  point = midpoint of range
previous cutoff point = minimum of range
```

```
while (not exactly one cell responds)
    search for cells larger than the cutoff point
diff = abs (cutoff point – previous cutoff point)
previous cutoff point = cutoff point
if (no cell responds)
        cutoff point = cutoff  point – (diff / 2)
else if (more than one cell responds)
        cutoff point = cutoff point + (diff / 2)
```

5.1.4 Stack

A stack is last-in first-out data structure. The operations are push, which adds data to the top of the stack, and pop, which removes the data at the top of the stack.

The following standard stack algorithms use a linked list of nodes as the stack. Each node in the stack contains data and a next pointer, which points to the next node in the stack. Top is a pointer that points to the node on the top of the stack. If the stack is empty, top is null.

```
push
        create a new node for the stack
        node->data = data
        node->next = top
        top = node
```

```
pop
    if  top is not null
        temp = top
        top = top->next
        delete node pointed to by temp
```

The associative stack algorithm below, from Potter[GEN7], treats the stack as a two-dimensional table. Each entry in the table contains the order number and the data. The first piece of data added to the stack has order number 1, the second has order number 2, and so on.

```
push
      find_max (order_number)
      write (max + 1, data) into
          unused memory
```

```
pop
      find_max (order_number) and
          return associated data
      mark memory as unused
```

The associative algorithm is much simpler because no pointers are used.

5.2 Database Algorithms

Since associative memory supports databases so well, database algorithms are of particular interest. This section discusses the typical relational database algorithms query and join. An introduction to relational databases is provided in Appendix A that defines the terms used below.

5.2.1 Query

A query is a common relational database application that is a combination of standard relational operators projection and selection. A record is chosen from the database because one or more

fields in the record matches the query criteria. Consider the following example table containing employee information.

Record	ID Number	Name	Start Date
1	1234	Smith, Joe	1-25-90
2	4567	Brown, Sue	3-30-81
3	2345	Jones, Fred	12-3-98
4	8765	Smith, Ann	5-1-93

A query for the ID Number of employees named "Smith" will display the ID Numbers from records 1 and 4. A query for the names of employees who started before 1991 will select the Names from records 1 and 2.

Assuming the table has a size of n records, the following standard query algorithm has a time complexity of O(n).

```
for (each record in the table)
    if (field(s) in the record match the query)
        select this record
```

In an associative memory, one entire record is usually stored in one wide memory word. Record fields of interest are isolated by using the mask register to mask out all other bits. The associative query algorithm has a time complexity of O(1) because the comparisons with the records in the table can all be done in parallel.

```
if (field(s) in any record match the query)
    select those records
```

5.2.2 Join

Join is also common relational database application. In its simplest form, a join uses two tables. If the contents of one field of a record in one table are the same as the contents of a field in the second table, the records are associated with each other. In the following example, record 2 in table A is joined to record 3 in table B because the employee ID number is the same.

Record	Name	ID No.
1	John	1234
2	Lee	4567

Table A

Record	ID No.	Rate	Hours
1	1111	$7.85	20
2	2222	$10.40	40
3	4567	$12.31	40
4	5678	$6.65	25

Table B

Assuming both tables are of size n, the following standard join algorithm has a time complexity of $O(n^2)$.

```
for (each record in first table)
    for (each record in second table)
            if (join field in first table = join field in second table)
                add a link
```

Assuming links can be written in parallel, the associative join algorithm has a time complexity of O(n) because the comparisons with the records in the second table can all be done in parallel.

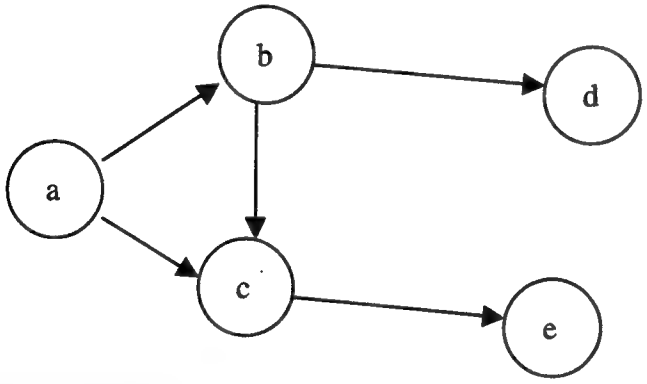
```
for (each record in first table)
    if (join field in first table = join field in any record in second table)
        add a link
```

5.3 Symbolic Processing Algorithms

An associative processor can be used for other kinds of processing than numerical calculations and database operations. As an example, a graph problem is discussed below.

5.3.1 Graph Connectivity

A graph consists of nodes and connections between the nodes, as shown below.



The specific problem is to find whether a path exists between two nodes, such as nodes a and e in the figure above. Assume that the graph is stored as an unordered table containing the source node and destination node for each connection. The table for the above graph is:

Source Node	Destination Node
a	b
a	c
b	c
b	d
c	e

The algorithm used by standard and associative memory is identical and is shown below [GEN2]. However, the time for one search of the table in standard memory has a time complexity of O(n), where n is the number of connections. The time complexity of the algorithm is therefore $O(n^2)$ in standard memory. The time for one search of the table in associative memory is O(1) and the time complexity of the algorithm is O(n).

The start node is e and the end node is a if this algorithm is used on the graph above.

```
current node = start node
previous node = start node
while (not done)
      search table for an unused entry where current node is the destination
      if (no entry found)
            current node = previous node  /* back up  and try another path */
            if (current node is start node)
                  done -- no path
      else
            mark this entry as used
            previous node = current node
           current node = source node from this entry
           if (current node = end node)
                 done -- path found
```

5.3.2 Image Processing Algorithms

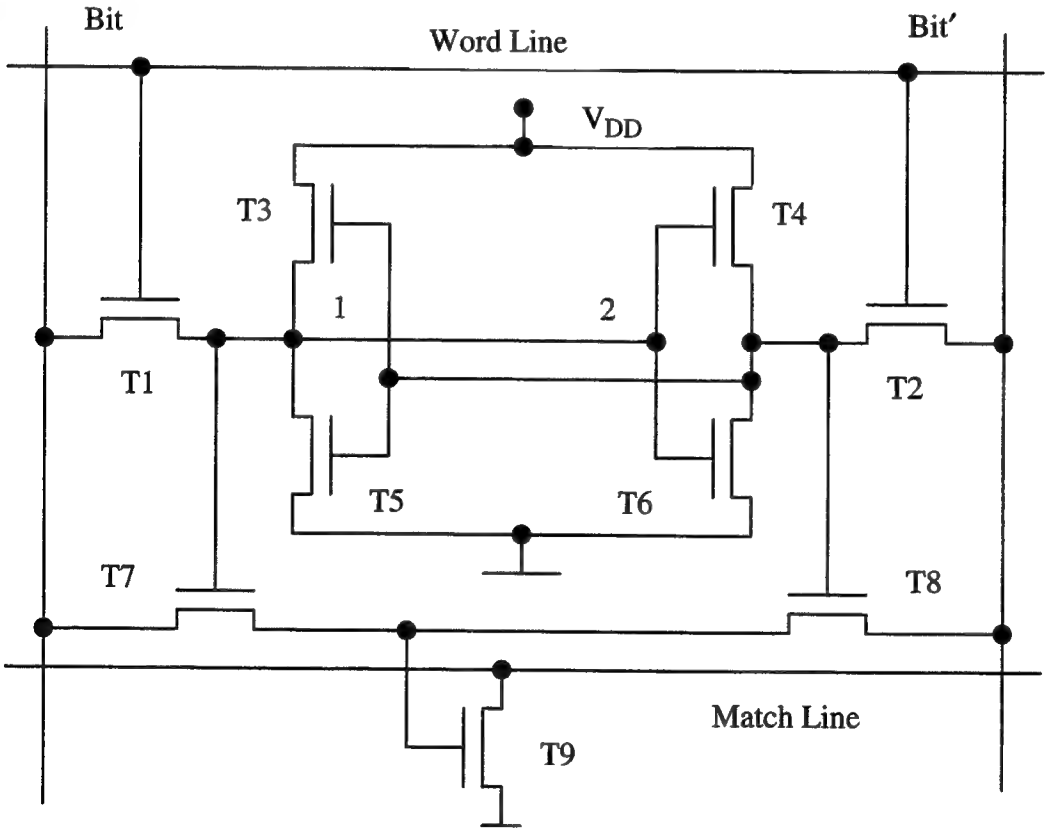
Image processing and computer vision applications are discussed in section 9. A specific set of image processing algorithms for a particular architecture are contained in Appendix II.

6 Semiconductor Technology

As noted in the proposal for this study [GEN6], associative memories can be implemented using various technologies. This section briefly discusses semiconductor technology. Optical technology, being less familiar to many readers, is discussed in greater depth in section 8.

6.1 Static Memory Cell

Fully-associative memory bit cells are typically extensions of a static RAM cell. The sample design from Grosspietsch[GEN3] shown below contains a standard six-transistor flip-flop and three additional transistors to implement the match logic.



Transistors 1 through 6 are a standard static RAM flip-flop. Transistors 7 through 9 implement the match logic.

The match logic works as follows, assuming the search bit is S. The match line and the word lines are first charged to high. Next, the Bit′ line is set to S and the Bit line is set to S′, which is the inverted version of S. If a high is stored in the flip-flop, node 1 is high. A low is propagated from Bit through T1 and T7 to the gate of T9. T9 remains off and the match line remains high.

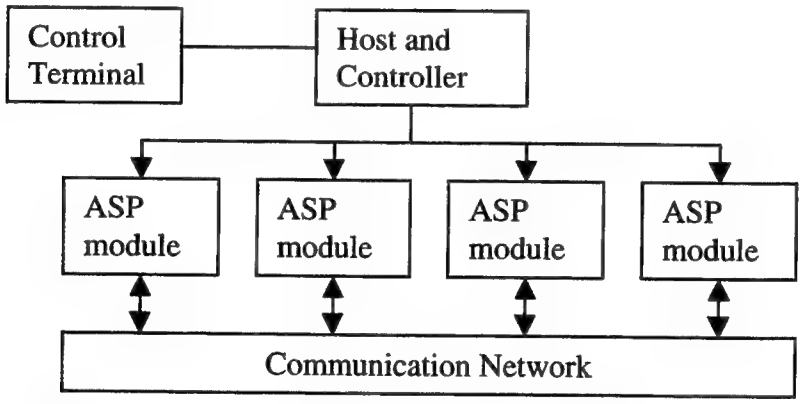
A mismatch opens T9, driving the match line low. Driving both Bit and Bit' low masks the bit cell.

6.2 ASP – Associative String Processor.

The Associative String Processor (ASP), a recent commercial development project, is implemented using semiconductor memory and includes all of the common associative processor features described previously. ASP is being developed by Aspex Microsystems Ltd UK and is based on research performed at Brunel University in London, England. The name of the system comes from the fact that it processes everything as a string in associative memory.

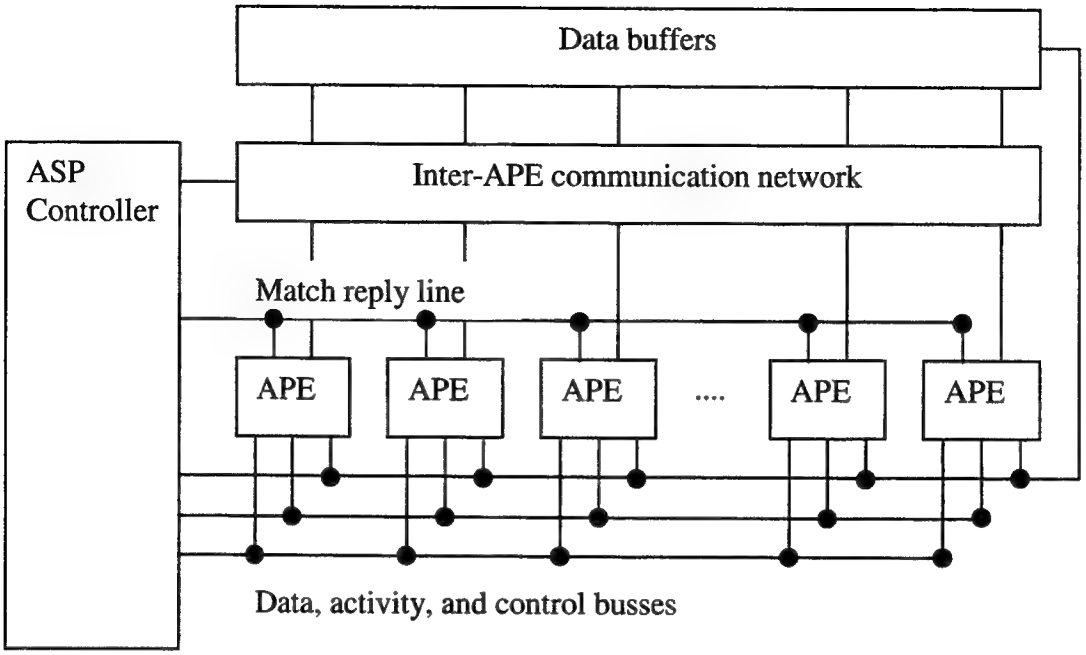
The following description is from Lea [GEN5] and Sima [GEN8].

The top-level architecture of an ASP system is shown below. ASP modules are connected together through a general-purpose communication network. The system is highly flexible, so the number of ASP modules and the communication topology can be tuned for specific applications or general-purpose computing.



Each ASP module contains several ASP substrings and the associated data buffers and control logic.

The simplified architecture of an ASP substring is shown below. The main components are Associative Processing Elements (APE). Each APE contains a word of memory (word widths are between 32 and 128 bits), a comparator, a full-adder, and control logic.
The data buffers permit either a slower serial APE-by-APE data exchange using the data bus, or a faster parallel data exchange with all APEs at once.



7 Optical Technology

The major attributes of optics that favor it in associative processing are high speed, large capacity, massive parallelism and noninterference of light beams. Optics can and have been exploited well in the areas of storage, information processing and interconnection. Free-space optical interconnection methods, while not yet a mature technology, are expected to resolve the wiring problems inherent with broadcasting the key word to all memory words using electronic technology.

This section of the report discusses optical devices, optical storage systems, and optical associative processing. Then, the report discusses some applications of optical associative processing. Irakliotis[] identifies two classes of optical associative processing. The first class deals with formatted (digital) data, such as relational database operations. The second class deals with unformatted (analog) data, such as image processors. The report provides examples of each.

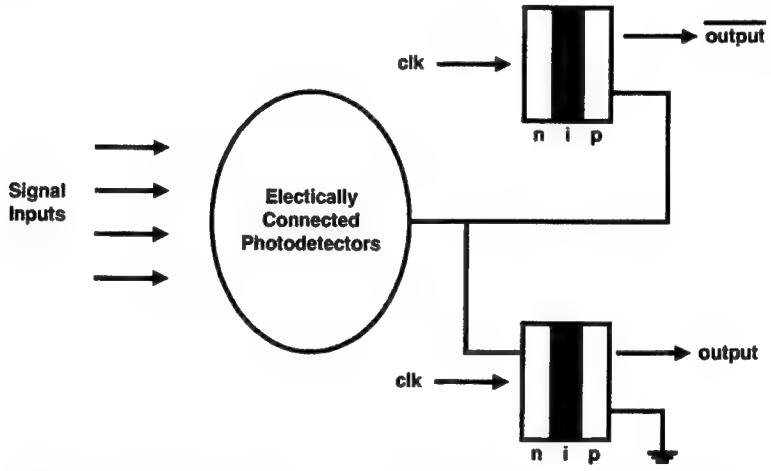
7.1 Optical Devices

Optical implementation of Associative Processors requires a number of components and devices. Such components include optical storage systems, SEED (and its variant the S-SEED) and Spatial Light Modulators (SLM). The next sections of this report review the above-mentioned components briefly.

7.1.1 Self-Electro-optic Effect Devices (SEED)

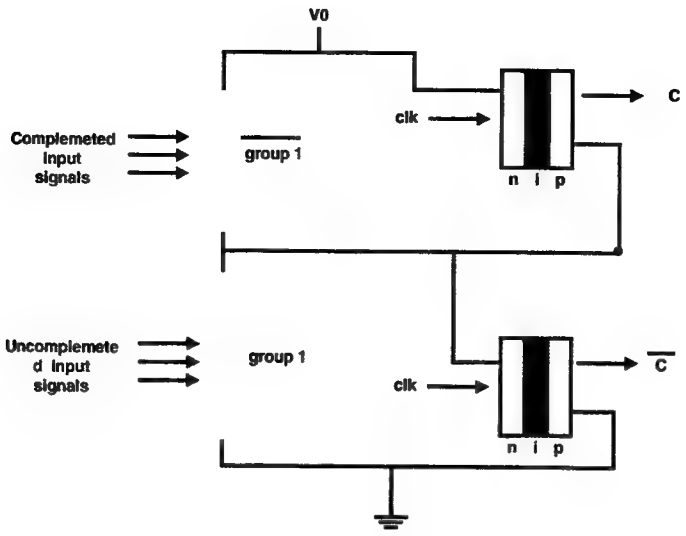
Research on optical digital systems has concentrated more on the use of 2D arrays of optical gates. The SEED is a leading candidate for such optically interconnected architectures that uses 2D arrays of optical or optoelectronic gates. These devices are differential in nature, and the logic states of the inputs and outputs are defined by the ratio of optical powers in a set of two beams. SEED technology is based on multiple quantum well (MQW) modulators, which consist of thin, alternating layers of narrow and wide bandgap materials such as GaAs and AlGaAs [Seed2]. Changes in voltage applied across the quantum well layers causes a significant shift in the distinct peaks of the absorption spectrum, thereby switching the well between absorptive and transmissive states. This electro-absorption mechanism is called Quantum confined Stark Effect [Seed2]. By placing multiple quantum wells in the intrinsic region of a reverse biased p-i-n diode the resulting device can modulate light in response to a change in voltage [SEED3]. This same device can also detect light.

7.1.1.1 Logical SEED (L-SEED)



Logic SEED Schematic Diagram. For differetial devices the topology of the connectios between the diodes is identical to CMOS circuits. The Electrically connected Photodetectors are also quantum well p-i-n diodes [SEED1]

The basic operation of an L-SEED is illustrated in the figure above. It consists of a group of input diodes with incident input signals and a pair of output diodes that modulates a pair of equal power clock beams to provide the output. The clock is basically used to amplify the output of the group of diodes.

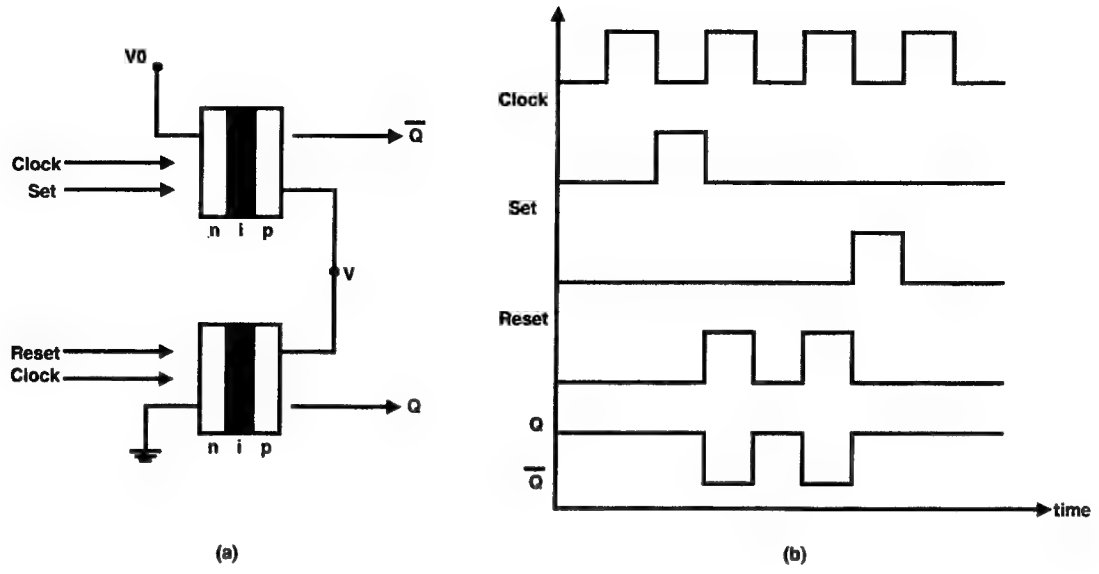


Differential L-SEED [SEED1].

The figure above shows Differential L-SEED, which consists of two groups of p-i-n diodes, namely the complemented group and the un-complemented group. These two groups are similar to PMOS and NMOS transistors in CMOS circuits. (One group is the dual of the other). As can be seen from the diagram, each differential L-SEED circuit has two outputs. The output is determined by the relative magnitude of the two beams. If the power of the uncomplemented beam is greater than that of the complemented beam, then this input is defined to be logic one and vice versa.

7.1.1.2 Symmetric SEED (S-SEED)

This is the optical version of an SR latch. It consists of two MQW pin diodes, connected in series. A voltage V (volts) is applied across this series combination. The two diodes are biased by complementary optical pulses (γ set,γbar reset) representing the incoming data. If the set input is greater than the reset input, the current in the corresponding (set) diode becomes greater than that of the reset diode. This pulls the set-well in to absorptive state and pushes the reset-well to transmissive state. The reverse happens when reset input is greater than the set input. Hence low switching intensities can change the state of the device from set to reset. After the device is put in to proper state, the clock beams are applied to the two diodes. The ratio of the clock beams applied to the two wells is equal to one. The clock pulses will amplify the state of the device and transmit it to the next stage of the system. In an array of S-SEED any one S-SEED can be optically or electrically enabled. Optical enabling is done by either applying or blocking the clock pulse. A Spatial Light Modulator can be used to select which S-SEED receives the clock pulse. By controlling the voltage applied to the diodes the S-SEED can be enabled electrically. If no voltage is applied both wells become absorptive, preventing the stored information from transferring to the next stage.



S-SEED device operating as a clocked SR latch.
(a) Schematic Diagram, (b) timing diagram.

The S-SEED is also capable of performing optical logic functions such as NOR, NAND, OR and AND. A method of achieving this can be found in [SEED3].

7.1.2 Spatial Light Modulators (SLM)

Light modulators change one or more properties of light across a surface or a spectral distribution. A Spatial Light Modulator is a device which, when optically or electrically loaded with an image or data page is capable of modulating transmitted or reflected light, with respect to the spatial distribution of their elements. The properties of light that could be modulated by SLMs are amplitude (or intensity), phase and polarization. This ability of SLMs to spatially or temporally modulate 2D optical wavefronts gives optics a degree of parallelism not found in electronics. SLMs were initially considered as transducers to input information to an optical processor, but it turns out that they can do much more complex functions such as analog multiplication, analog addition, signal conversion (power amplification, electrical to optical) [SLM1]. SLMs can even be considered as storage units. Although some SLMs exhibit long-term storage capability, SLMs can be treated as a buffer storage element between an optical computer and an optical storage unit (such as optical disk, Page Oriented Holographic Memory) permitting parallel inclusion of stored data into the optical computer.

7.1.2.1 Classification of SLMs

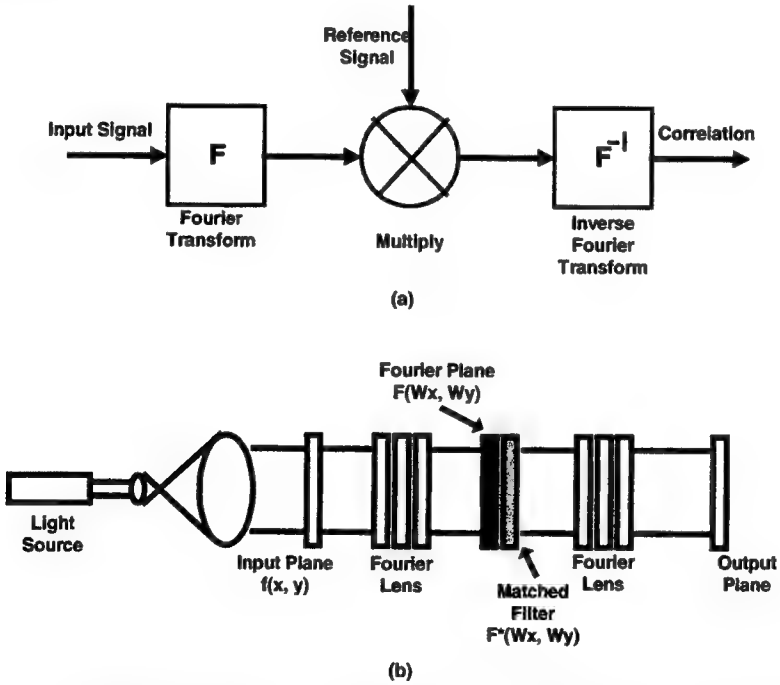
Although SLMs can be classified in a number of modes, one basic mode of interest is the Addressing mode. In this mode, the SLMs are classified as electrically addressed (ESLM) and optically addressed (OSLM) SLMs. The input signal for an ESLM is electrical. This electrical signal modulates the readout light, thereby transforming an electrical signal in to an optical signal. The input signal to the OSLM is optical signal, which modulates another optical signal.

The ESLMs serve as an interface between the electronic and the optical part of the processor, while the OSLM acts as an optical I/O device or as an active element in an optical processor.

7.1.2.2 Applications of SLMs

7.1.2.2.1 Optical Correlators

Correlator is a processor that compares (looks for a match) between two patterns. The figure below shows an optical frequency plane Correlator. As explained in the figure, if there is a match between the input and reference (stored) patterns then an amplitude modulated plane wave will result from the multiplication. Another Fourier transform on this would now result in an autocorrelation function with a sharp peak. When the input and the reference patterns do not match, a cross-correlation occurs. This cross-correlation contains no sharp peaks. For partial matches between the two patterns, the correlation will still be indicated by a bright spot of light in the output plane. In this case, the location of spot indicates the relative location of the matched region in the input plane.



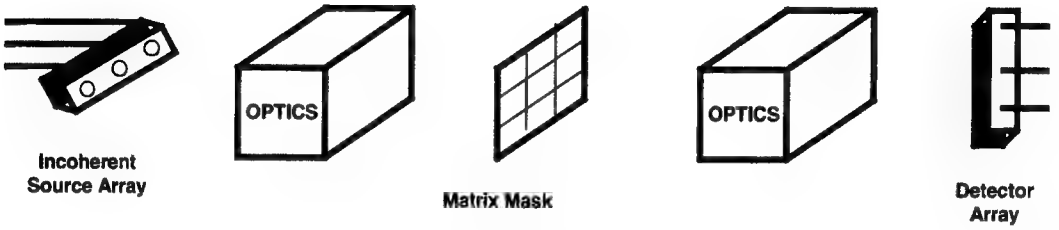
a) Simple block diagram of a fourier correlator, b) Block diagram of a frequency plane correlator. The correlator performs the fourier transform of the "complex amplitude distribution occuring in the Input Plane" and multiplies this with the stored fourier transfrom in the Fourier Plane. If there is a match between the input and the stored transform, then the beam emerging from the filter will be well focussed at the Output Plane[SLM1].

The use of SLM in the above Correlator is as an input plane device, permitting thousands of 2D data patterns to be searched each second for a given pattern. The Correlator can be made even

more powerful by using an SLM in the Fourier plane, permitting a large number of search patterns [SLM2].

7.1.2.2.2 Optical Matrix Multipliers:

Matrix operations such as vector-matrix and matrix-matrix operations are perfect candidates for 2D optical processors. A vector-matrix multiplier is shown in the following figure.



Vecor-Matrix Multiplier.

Vector values are represented by levels are intensity of the linear source array, while matrix values (mask) are encoded as levels of transmissivity of the SLM pixels. Each source element is spread over a column of the SLM pixels (mask). The resulting products are summed on to the detector. The spreading and collecting is done by spherical and cylindrical lenses. The spreading could also be done by holographic gratings, in which case one must use a coherent source array.

7.2 Optical Storage Systems

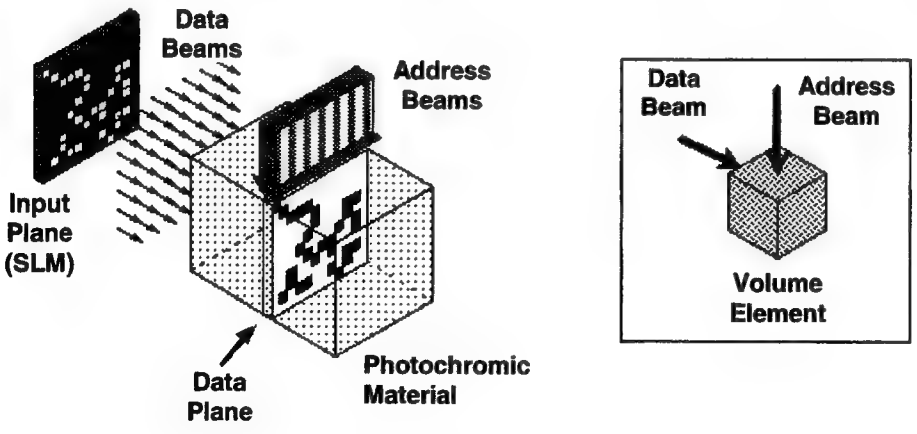
Processing of data in today's high performance computers are mainly limited by the data storage and retrieval rates rather than by the speed of the processor. Most applications that run on today's computers are input/output intensive and rely on the computer's ability to process a vast amount of data. This vast amount of data cannot fit in main memory but must be retrieved from a secondary storage. The performance of the secondary storage system is determined by the storage capacity (in bytes) and data transfer rates (in megabytes per second). Both of these quantities must be as high as possible. Storage media such as magnetic disk, electronic RAM are limited by their 2D nature. In general optical memories are much superior to electronic or magnetic storage in the sense that they have a high data density, high-speed parallel data access along with the ability of optical memories to allow data storage in all three dimensions. For optical memories, the density of stored information is proportional to the reciprocal of the wavelength raised to the power of dimension used to store information. Hence, an increase in dimension (from 2 to 3) would result in increase in the storage density. In contrast to the serial output of magnetic and optical disks, the output of 3D optical memories such as volume holograms and two-photon memories can be two-dimensional which, combined with the absence of moving parts in the read/write mechanism, results in higher transfer rates that can reach hundreds of megabytes per second. An additional advantage of holographic storage is its ability to operate in an associative addressing mode that is always preferable in a database environment.

The above characteristics make 3-D optical memories ideal candidates for database secondary storage.

7.2.1 Two-Photon 3-D memory devices

The two-photon 3D memory devices are based on two-photon absorption, which relies on the simultaneous interaction of two-photons inside a photochromic material causing it to undergo a structural change. The molecules of the photochromic material get excited from an unwritten ground state to an electronic state of higher energy by the simultaneous absorption of two photons. The first photon excites the molecule to a virtual state, while the second photon further excites the molecule to a real excited state. The energy required to reach this excited state is greater than the energy of either photon alone. Therefore, the corresponding molecule at points in the memory volume where both beams intersects gets excited to a higher electronic state. At all other points, the wavelength of each photon alone is longer, and hence corresponds to lesser energy. Therefore, these single photons are not absorbed and the beams propagate through the volume without being absorbed. Subsequently, the excited state decays into a different molecular structure, which becomes the written form of the molecule.

Data is stored in planes within the crystal. Each plane is a 2D array of volume elements each corresponding to a bit. A volume element is selected (addressed) by two intersecting beams. During write cycle, these beams are called data beam and address beam. The data beam originates from a SLM that holds the data page to be written in the memory. This beam is focussed by an autofocus mechanism (also called DFL – Dynamic Focusing Lens) on to the desired data plane. At the same time, the data plane is illuminated by a number of address beams. The two beams intersect in the volume element and cause the photochromic molecules to undergo structural changes. The figure below, illustrates the above mentioned write process.



Two-photon 3D memory system [BOOK].

The written molecules absorb light at longer wavelengths than the original molecules and are read by the fluorescence induced by two-photon absorption. Moreover, since reading is achieved at longer wavelengths, no writing is performed while reading. Erasing the information can be achieved by one of the following methods. By increasing the temperature of the memory device,

the written molecules are raised to an energy barrier separating the write and read forms causing the written molecules to return to the original form. However, this method does not provide selective erasure of specific bits in the memory. For selective erasure, we can irradiate the device with a specific wavelength other than that used for either reading or writing [2PHO4].

Architectures for optoelectronic associative processing have been proposed based on two photon 3D memories. It is important to note here that these processors do not offer full parallel associative processing. These processors are block-oriented processors in the sense that they do not allow searching in a single step, but through a sequence of steps.

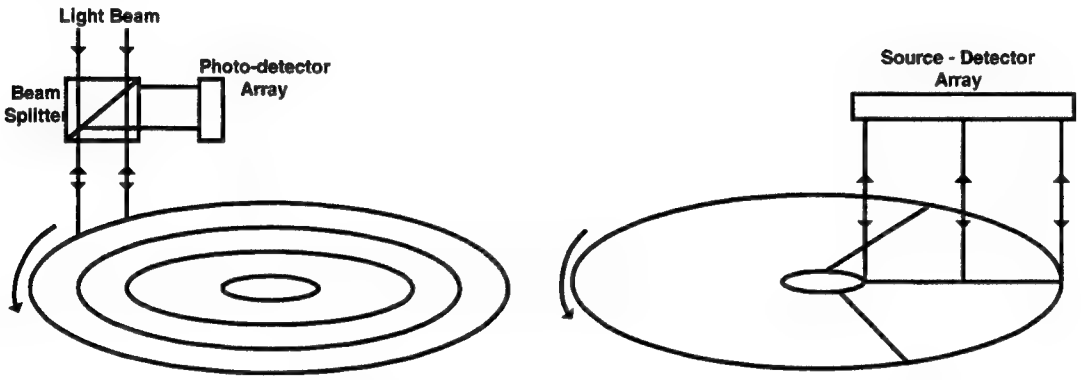
The following table from **[2pho3]** compares 2-photon 3D memory with other memory devices.

Device	Capacity	Cost Per Mbyte	Access Time	I/O Bandwidth	Volatile Storage	Comments
2-Photon 3-D Memory	$10^{11} - 10^{14}$ bits	$0.1-$10	1ms – 1us	1 Gbit/sec - 1Tbit/sec 10^6 Channels Parallel	No	Device at research/development stage.
Static RAM (BICMOS)	256 Kbit	$1000	10 nsec	100 Mbit/sec Serial	Yes	Expensive and fast memory used in caches and supercomputers.
Dynamic RAM (CMOS)	1 Mbit	$100	100ns	10 Mbit/sec Serial	Yes	Used for main memory. Beyond 64Mbit, a breakthrough in technology is needed.
Solid-State Disk Drive	1.28 Gbits	$300	200 us	24 Mbit/sec Serial	Yes	This memory fills the gap between main memory and disk drives
5.25" Optical Disk Drive	10 Gbits	$0.25	20 ms	20 Mbit.sec Serial	No	Density is expected to double every 2.5 years. Other parameters grow at slower pace.
5.25" Magnetic Disk Drive	10 Gbits	$1	10 ms	20 Mbit/sec Serial	No	Density is expected to double every 2.5 years. Other parameters grow at slower pace.
IBM 3390 Disk Drive	120 Gbits	$10	10 ms	36 Mbit/sec Serial	No	Used in Supercomputers. Trend is to replace these by 5.25" disk drives, since they are more cost effective.
Digital Tape Recorder	2000 Gbits	$0.0025	Sequentia l	150 Mbit/sec	No	Used for off-line data storage. Does not support random access.

7.2.2 Optical Disks

Optical disks offer high recording density and parallel access, reducing memory interface bottleneck as offered by current (serial) computing architectures. Both Write Once Read Many memory and magneto-optic read/ write disk drives are available for high-density storage on mainframes and personal computers. The conventional mode of both writing and reading in present day optical disks are serial. A laser will write one bit at a time on the disk. Reading is achieved by using lower power beam to illuminate the each of the bit individually, and based on

the reflected or transmitted intensity; the bit is decoded as a logical 1 or 0. Although optical disks favor serial readout, they are well suited for parallel readout as well. If the large portion of the disk is illuminated with a collimated beam, the reflected or transmitted light contains all the data recorded in the illuminated area, which can very well be detected by a detector array. This is shown in the figure below (a). This technique can provide data rates up to hundreds of Mbytes, but is not suitable for parallel recordings, since independent modulation of particular beam is not possible. Another method shown in the figure (b), uses 1-D semiconductor array of sources and detectors placed above the disk, and along a radial line, so the each track on the disk has its own source-detector pair. This method allows individual beam modulation and is suitable for recording as well.

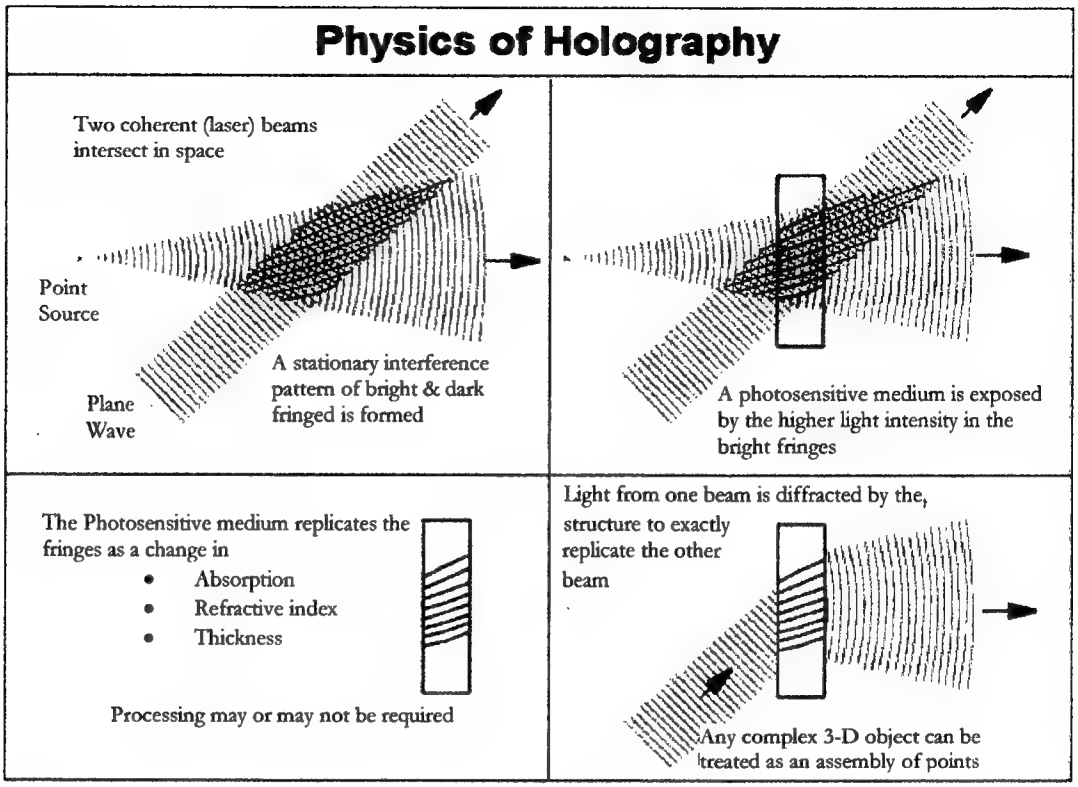


Multiple beam read from optical disk. a) Single laser beam expanded in to multiple parallel coherent beams, to fecilitate parallel readout. b) 1-D Source-dector array that fecilitates both parallel read and write [DISK3]

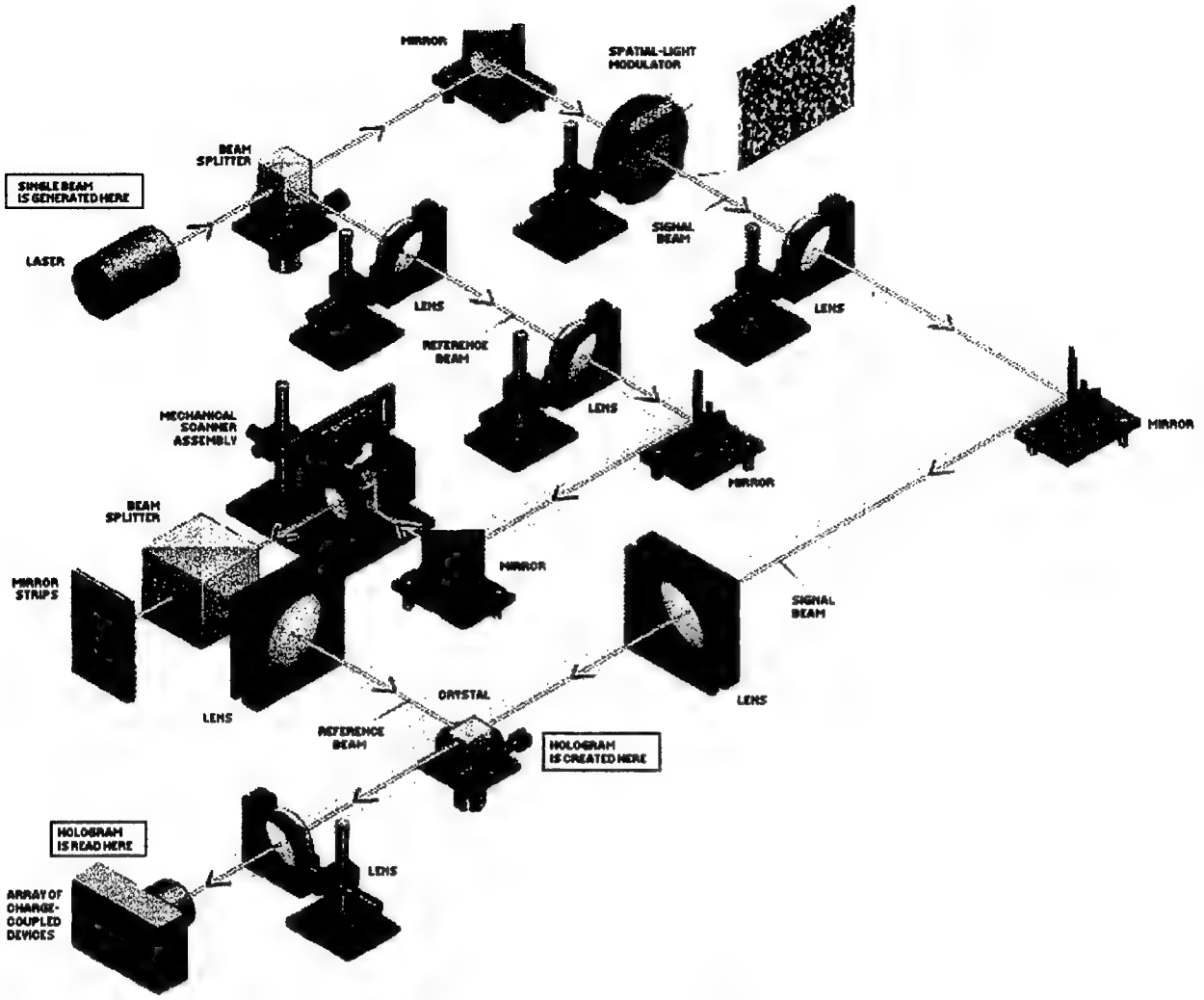
According to [DISK3], linear densities of 25 kbits/cm for optical writable disks and 10 kbits/cm for magneto-optical erasable disks can be achieved. With rotational speed of several hundred rounds per minute, about 5 x 10^5 records can be scanned in few tens of milliseconds.

7.2.3 Holographic Data Storage

In holographic data storage, a photosensitive medium (e.g. crystal cube or polymer film) is exposed to the interference pattern that is generated when an object laser beam, with the data encoded in it, is intersected by a second, coherent laser beam. The photosensitive medium replicates these interference fringes as a change in optical absorption, refractive index, or thickness. Data are retrieved from the medium by exposing it to light from just one of the beams. The figure below shows the physics behind holographic recording.



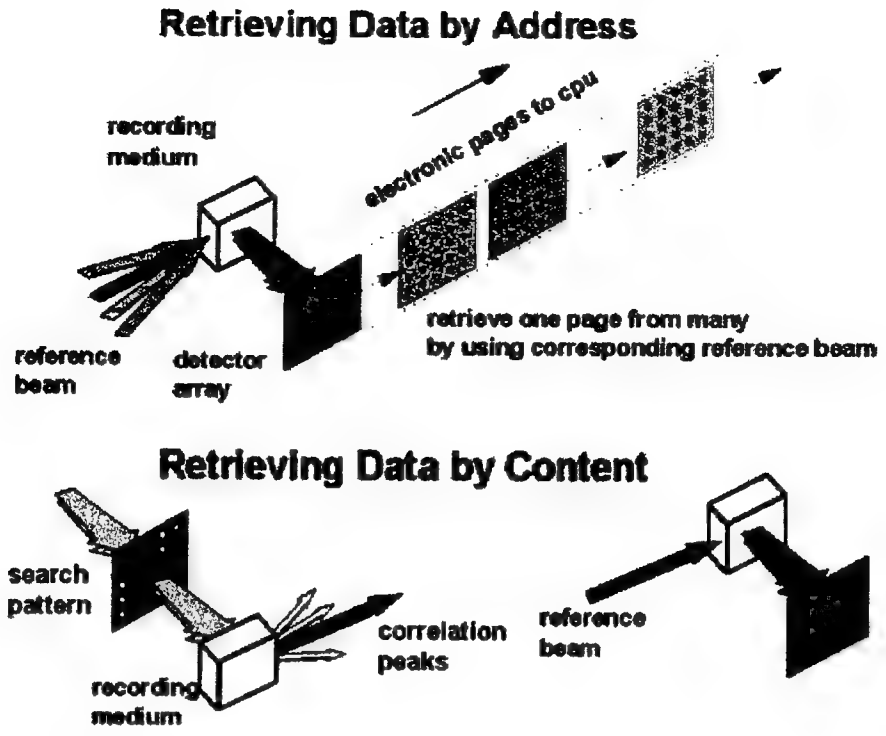
To encode the bits of a page in to the object signal, the bits must first be represented as a pattern of clear and opaque cells in a SLM. A laser is then shined on this pattern, which can now be focussed by lenses to create the object signal. A hologram of the page of data is created when this signal meets the addressing or the reference beam in the photosensitive medium. After a page is recorded in the crystal, the angle of reference beam can be increased until the reconstruction of first hologram disappears. A new page can now be recorded with this angle for the reference beam. This procedure called angle multiplexing can be repeated for recording several pages, the number of pages being limited by the dynamic range of the material. As more holograms share the same crystalline volume, the percentage of light diffracted by each hologram (during reconstruction) diminishes. The figure below is shows the recording of pages of data in a holographic crystal. The Spatial light modulator is loaded with the data pages. A combination of lenses, mirrors and mechanical scanner assembly generated the reference beam.



Recording of a photosensitive crystal (Lithium Niobate) with pages of data. [HOLO1]

7.2.3.1 Data retrieval from holographic memory

Once the hologram is recorded, either of the two beams (signal, reference) can be used to reconstruct the other beam i.e.; light from one beam is diffracted by the photosensitive medium, to exactly replicate the other beam



Address based and Content based retrieval of data.
[OCG website at Arizona State University - http://www.ece.arizona.edu/department/ocppl]

Address based retrieval

In this method, the object beam can be reconstructed by illumination of the hologram with the original reference beam. Lenses focus the data page onto a detector array where the bright and dark pixels can be converted back to digital data. When the hologram is stored throughout a thick storage material, Bragg diffraction causes the strength of reconstruction to be sensitive to changes in the angle of reference beam. This Bragg mismatch is most sensitive to angle changes in the plane formed by the object beam and the reference beam, so that we can store and independently address multiple data pages by merely steering the reference beam.

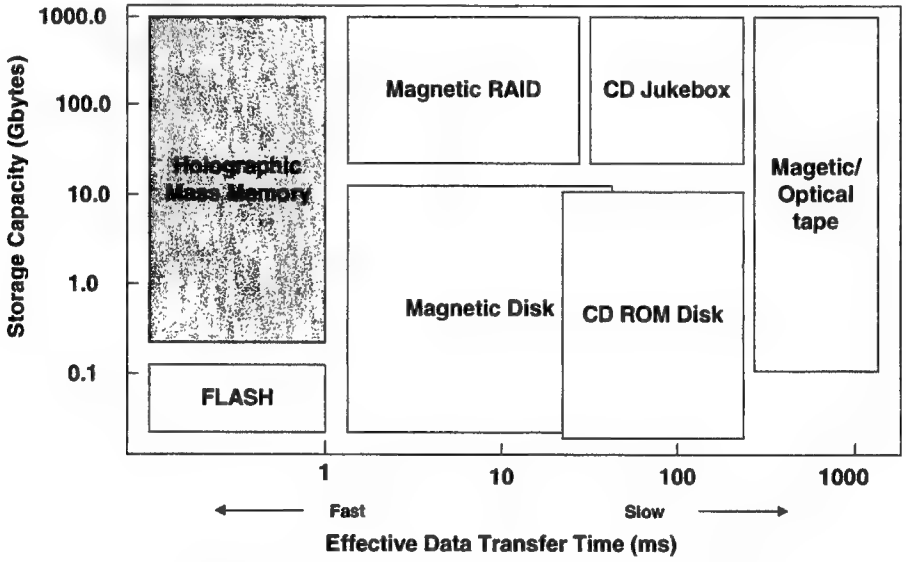
Content-based retrieval

Volume holograms, upon illumination with the object beam, reconstruct all the angle multiplexed reference beams that were used to record data pages in to the volume. The amount of power diffracted in to each output beam is proportional to the correlation between the input data page and the stored data page (recorded with that particular reference beam). Each output beam can be focussed on to a detector array to form a correlation peak. If the pattern that compose these pages correspond to the various data fields of a database, and if the stored page represents a data record, then the optical correlation has just compared the entire database with the search argument simultaneously. [HOLO2].

In recent experiments at Caltech, a volume holographic disk of LiNbO3 was used for data storage. In a volume of 2 x 2 x 5 mm^3, 100 holograms were recorded, each with 220 x 320 pixels [2PHO2]. Some of the characteristics of Holographic Storage is shown in the following table.

	Page	Volume
Storage Density	100 Mbits/Cm^2	100Gbits/cm^3
Write Transfer Rate	1 – 100 Mbits/sec	
Read Transfer Rate	0.1 – 1 Tbits/sec	

The following figure, from a Laser Focus World article (Vol 32, issue 4, April 1996), classifies different hardware approaches to data storage with respect to storage capacity and effective data transfer time. The effective data transfer time is a measure of the time required to fully retrieve a arbitrarily located data block from the system. It is a combination of the data transfer rate – the rate at which data are transferred from memory to the CPU and data latency – time lag between address setup and appearance of valid data at the output.



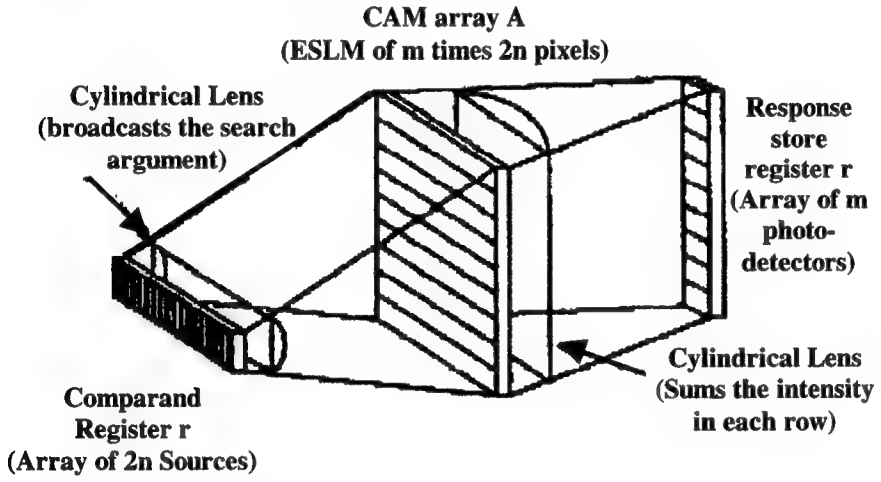
Classification of different hardware approaches to data storage w.r.t storage capacity and effective data transfer time [Laser Focus World - April 1996].

7.3 Optical Associative Processing and Processors

Most of the optical associative processors proposed to date are typically for relational database operations and image correlation, although there are a few papers on certain arithmetic processors. This section of the report summarizes a few of processor architectures that are based on the above mentioned optical storage systems. A review of the relational database operations is provided in Appendix 1.

7.3.1 One-Dimensional Optical Content Addressable Memory:

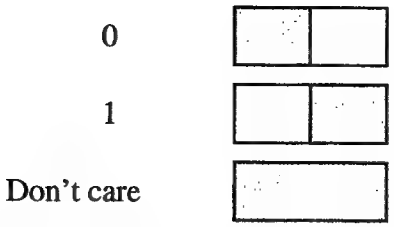
A 1-D OCAM consists of a 1D-source array, 2D memory (Usually an E-SLM), a 1D detector array connected to response register and necessary optics to do expand and sum operations. The



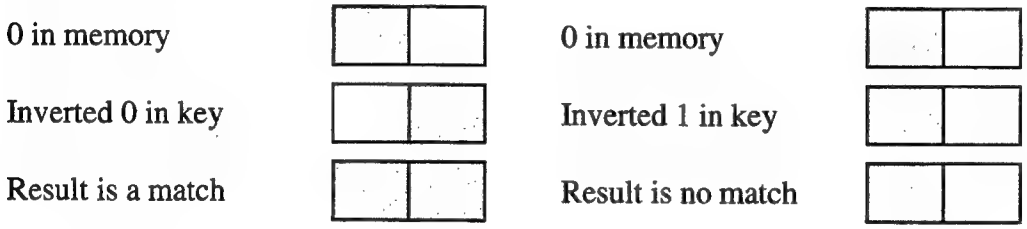
One-dimensional OCAM [CAM]

source array is m bits wide and consists of the search argument. These m bits are expanded to form 2D - m x n matrix where n is the number of words stored in the memory (E-SLM). Note that all rows of this matrix are equivalent. The E-SLM acts as the Content Addressable Memory. It can be loaded from a page oriented holographic memory or any other 3-D memory with parallel data retrieval, as will be seen later. As soon as the 2D-search argument is imaged on to the SLM, the SLM does the correlation operation. The results of this correlation operation can be detected by the 1-D detector array, which in turn stores the results in the response register. Two cylindrical lens can be used – one to expand the 1-D search argument and broadcast it to the E-SLM, and the other to sum (logical OR) the intensity in each row of E-SLM after comparison.

Akyokus [DATABASE1] uses a spatial encoding scheme where each bit of information in the source array and the memory array requires a combination of two pieces of optical data.



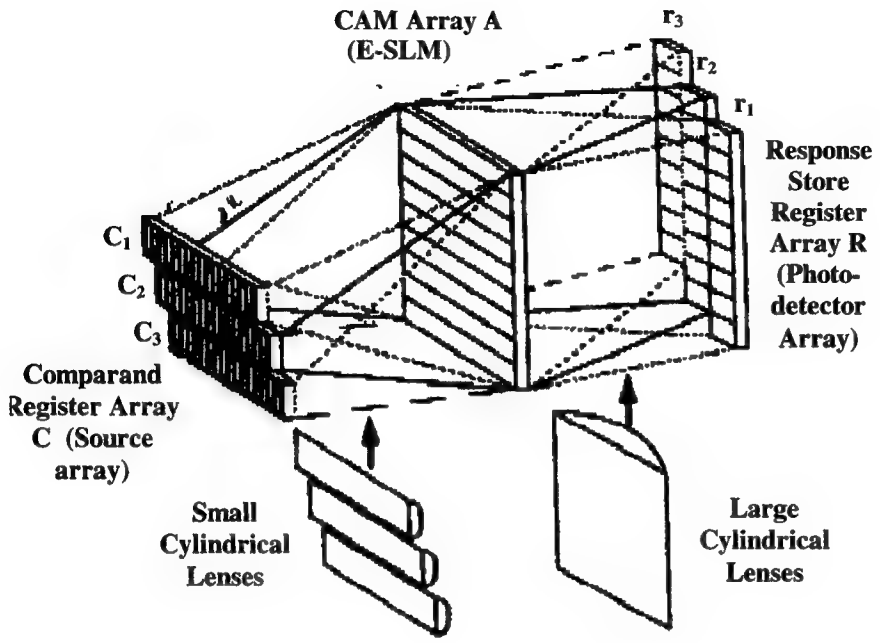
To perform a comparison, first invert the 1's and 0's in the key word before putting it in the source array, so that a 1 is encoded as a 0 and a 0 is encoded as a 1. The optical combination of the key bit and the memory bit will then be as shown below.



If the combination of an entire key word and an entire memory word results in a dark strip, the detector after the second lens will be dark, or 0, and a match will be detected.

7.3.2 Two-Dimensional Content Addressable Memory:

A CAM with multiple search capabilities is called as a 2D CAM. The search argument in this case is a 2D-source array. The 2D OCAM consists of a 2D-source array, 2D memory, 2D-detector array and other necessary optics.

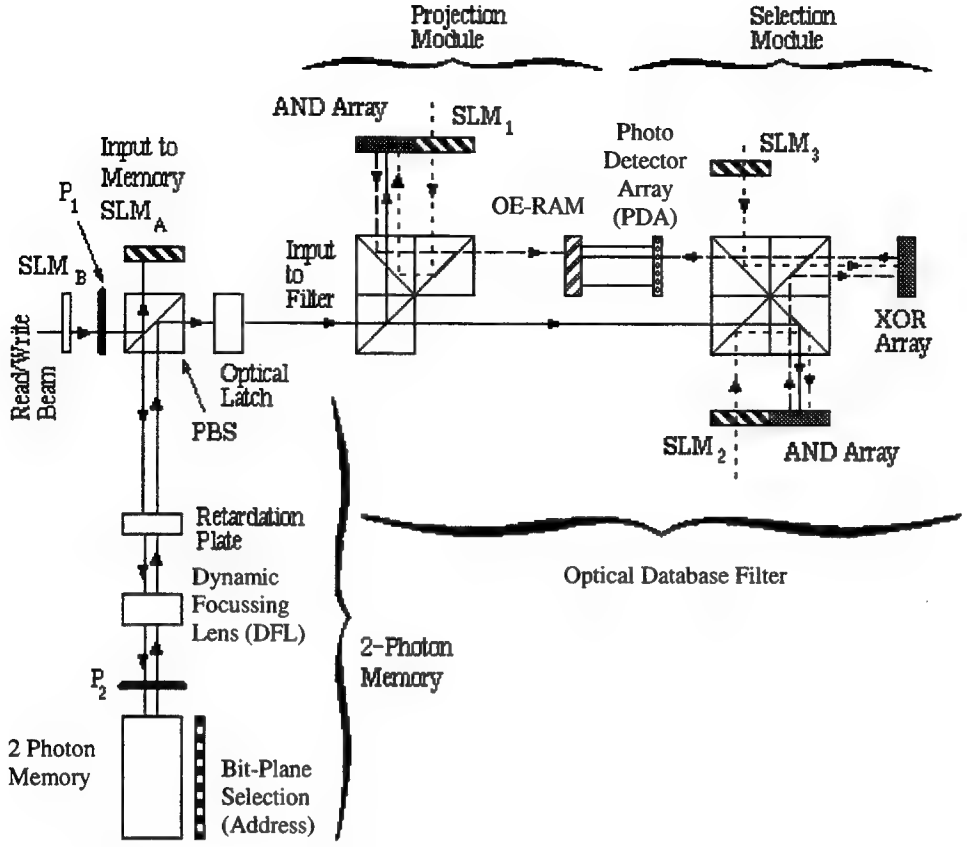


Two Dimensional OCAM [CAM]

A 1D CAM can be used in relational database operations, where a tuple (Appendix 1) is coded in 'm' bits. A 2D CAM can be used either in relational database operations or as image correlators. Sometimes, special encoding schemes are used to encode the tuple of a relation onto a whole page, in which case 2D CAM must be used.

7.3.3 Associative Processors based on two-photon Memory

As mentioned earlier, two-photon Memory based associative processors are block-oriented processors. Associative searching is not possible in this type of memory. This type of memory is aimed at reducing the data transfer rates from a parallel optical memory to an electronic host, by performing on-the-fly selection and projection operations [Appendix 1] on 2D pages of relational data. The figure below illustrates a system that combines a two-photon memory with an optoelectronic database filter.



Two-photon memory with optoelectronic database filter [2PHO2]

As seen from the figure, the optical database filter consists of a projection module and a selection module. The two-photon memory output is a 2D page with one record per row. This output is split in to two and fed in to the two modules of the optical filter. The projection operation requires masking out the undesired data fields from the 2D page. This is done by AND-ing the data page with the projection argument (user-defined). In the above system, the projection argument is given via SLM1. The output of the AND array is fed to an optoelectronic array (OE RAM) which accepts optical input and stores it in an electronic form into a RAM type memory [OECMOS]. In the selection module the data-page is first AND-ed with a selection mask from SLM2. This step is equivalent to the projection module. The output of the AND gate is now directed to an XOR array by the beam-splitter, to be XOR-ed (compared) with the selection argument which is replicated on to each row of the matrix in SLM3. A cylindrical lens is now used to OR together the results of XOR operation in order to determine the matching rows which in turn is detected by the linear photodetector array (PDA). The following equation describes the selection operation:

$$\begin{vmatrix} h_1 \\ h_2 \\ \vdots \\ \vdots \\ h_m \end{vmatrix} = \begin{vmatrix} d_{11} \oplus S_{11} + d_{12} \oplus S_{12} + \ldots + d_{1n} \oplus S_{1n} \\ d_{21} \oplus S_{21} + d_{12} \oplus S_{12} + \ldots + d_{1n} \oplus S_{1n} \\ \\ \\ d_{m1} \oplus S_{m1} + d_{m2} \oplus S_{m2} + \ldots + d_{mn} \oplus S_{mn} \end{vmatrix}$$

where,

h_i = elements of the vector identifying matches

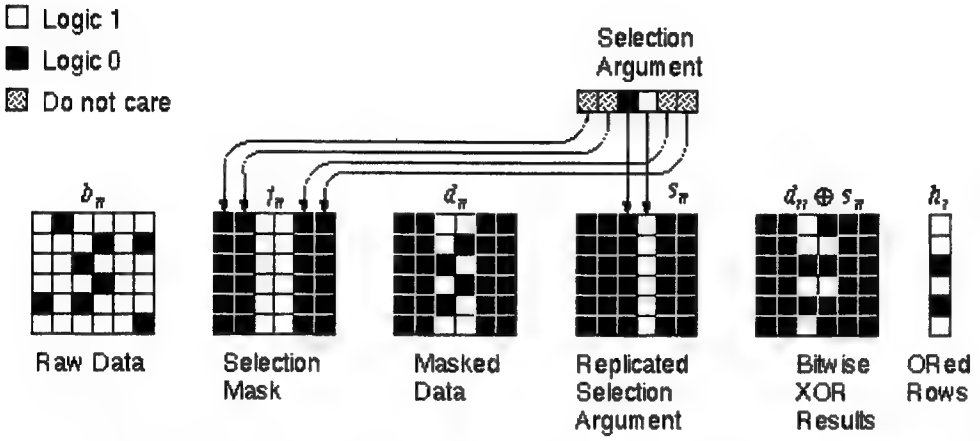
d_{ij} = b_{ij} AND f_{ij} (result from the AND array).

f_{ij} = bits of the selection mask (from SLM2).

b_{ij} = data bits from two-photon Memory.

S_{ij} = bits of the selection argument (from SLM3).

Note: $S_{1j} = S_{2j} = ... = S_{mj}$, j = 1,2, ... ,n, since the selection argument is replicated on to each row of the SLM3 matrix.



Block-oriented associative processing for Optoelectronic Data Filter. This figure illustrates the selection operation. [BOOK]

The selection mask allows selection based on only specified attribute. When selection and projection operations are performed simultaneously, the PDA determines which rows of the OEROM will be readout and subsequently transferred to the electronic host computer. The operation modes of the filter are summarized in the following table.

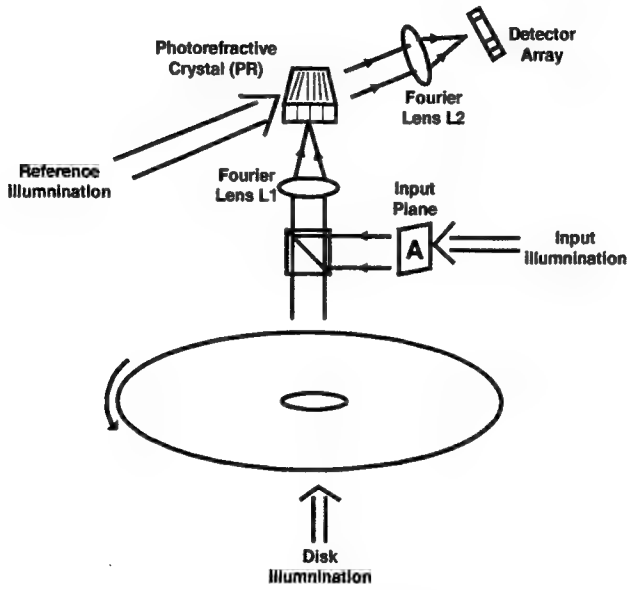
Operation	SLM1	SLM2	SLM3
Unconditional Retrieval	All active	Inactive	Inactive
Projection	Active	Inactive	Inactive
Selection	All active	Active	Active
Selection and Projection	Active	Active	Active

7.3.4 Associative Processing with Optical Disks

7.3.4.1 Closed loop optical disk based associative memory [DISK2]:

This optical disk based associative memory is an image correlator with electronic feedback circuitry. As mentioned previously, correlator is a processor that compares (looks for a match) two patterns. In this system (see figure below), the optical disk contains the reference library images. To initiate the correlation process, a holographic crystal (PR) is first recorded with the

Fourier transform hologram of the input image by illuminating the crystal with the input and reference illumination. The input illumination comes from SLM interfaced to a video camera. The Fourier lens L1 between SLM and the crystal takes the Fourier transform of the input image. The hologram thus recorded in the crystal can be read by using the reference library from the disk. To do so, the input and reference illumination is turned off and the disk illumination is turned on. The product of the input (stored in PR) and reference Fourier transform (from the disk) takes place at the crystal. This is then inverse Fourier transformed at Fourier lens L2 to yield the correlation output. More details on this correlator can be found in [DISK2].

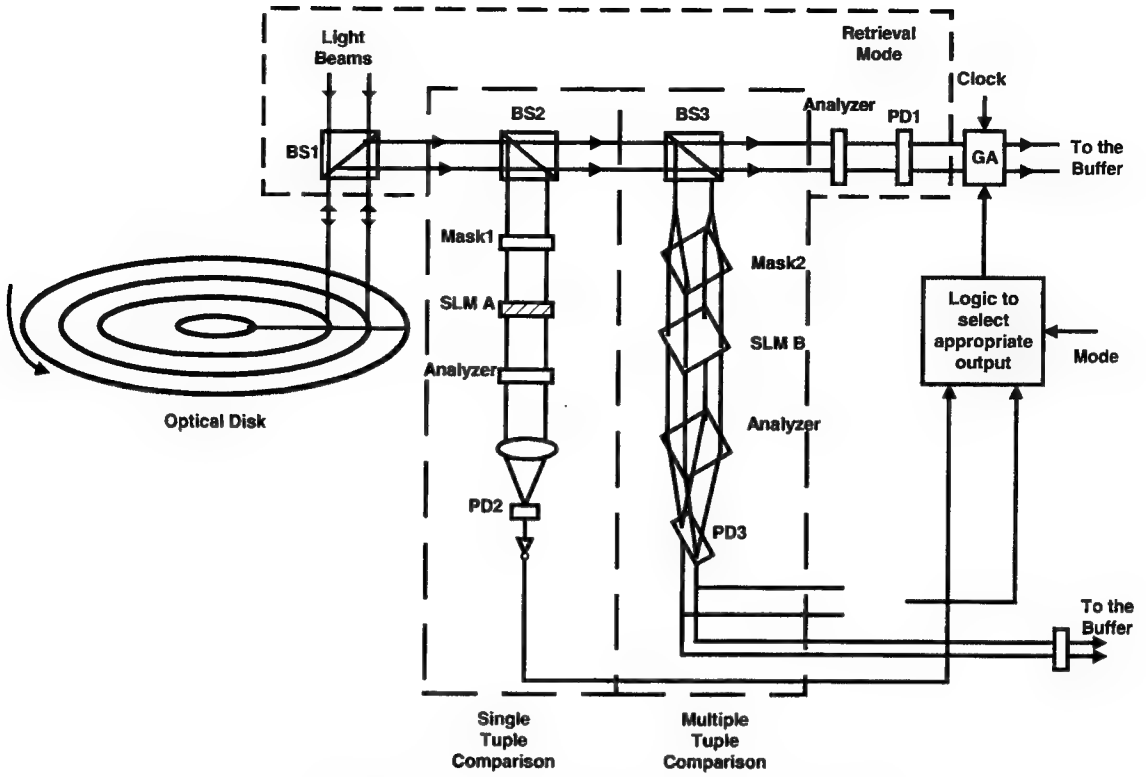


Optical disk based image associative memory [DISK2]

7.3.4.2 PHOEBUS - An optoelectronic database machine based on parallel optical disk [DISK3]:

In this database machine, large magneto-optical read/write disks hold the database. A tuple of a relation is recorded along the radial dimension of the disk - across multiple tracks. A large optical disk can have tens of thousands of records. If the number of bits per track is less than the number of tracks, then an entire record can be read in parallel using either the two methods described earlier.

The Optical Database Processor (ODP) processes the data (tuples) received from the Optical disks. It can operate in three different modes – Retrieval mode, Single Tuple mode and Multiple tuple mode. These three modes are explained below with reference to the following figure.



The optical database processor in PHOEBUS [DISK3]

Retrieval mode:

In this mode, data from the disk is read directly in to a buffer. The arrangement is similar to that explained in section 7.2.2 where, multiple tracks are read in parallel by an equal number of polarized light beams. The reflected beam is directed to a photodetector array through the beam splitter on to a 1D-photodetector (PD_1) array, which has one cell per track. Note here that the data is read in a radial direction across parallel tracks. The output of PD_1 is then directed in parallel to an electronic buffer through gate array (GA). GA is used to select the output from the three possible modes of operation (Retrieval, single-tuple-comparison and multiple-tuple-comparison)

Single-Tuple-Comparison mode:

In this mode, a tuple is read in parallel from the disk and passed on to the GA as well as a mask through a beam splitter. The mask is used to block the undesired bits from being compared. The comparison takes place at the SLM following the mask. The SLM holds the reference pattern. The result of the comparison is ORed (using a lens) and passed on to another photodetector PD_2, which in this case is a single cell. A zero in this cell indicates a match. The GA passes the tuple that was initially read form the disk if and only if there was a match. This mode is used for the selection (Appendix 1) operation.

Multiple-Tuple-Comparison mode:

This mode is used if each tuple must be compared to a number of different patterns. In this mode, the readout tuple from the disk is expanded in to a 2D matrix before reaching the SLM. All the rows in this matrix contain the same tuple. This is done by expanding the readout beam along one direction, using a cylindrical lens. Now, this 2D matrix is directed to a 2D SLM. Each row of the SLM contains a different record (or tuple). The readout pattern is compared with each of these records and the result is now stored in 1D PD_3. Even a single zero in PD_3 (indicating that there was a match) enables the GA to pass the tuple to the readout buffer.

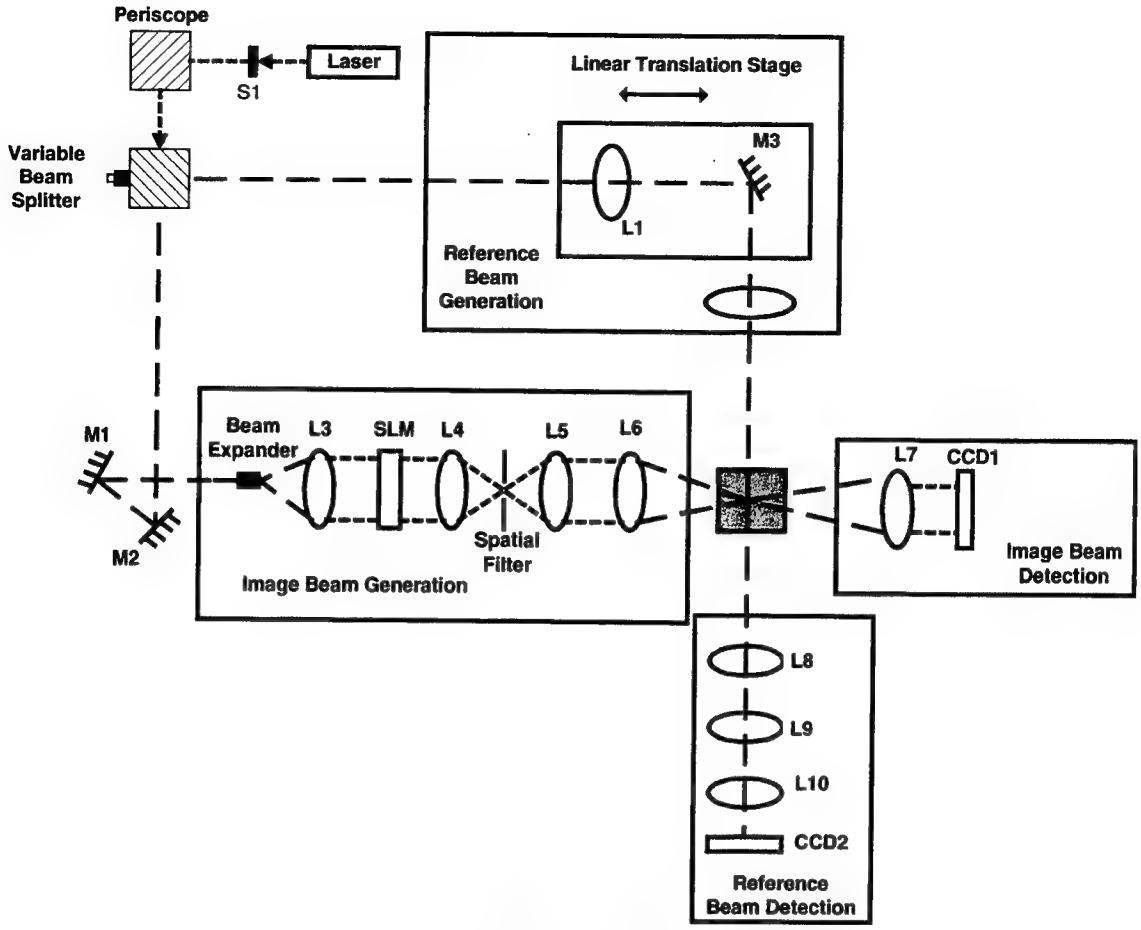
A combination of the above three operations along with some electronic control is used to perform the relation database operations such as selection, join. For more details, please refer to [DISK3].

7.3.5 Associative Processing with Holographic memory

7.3.5.1 Volume Holographic Storage for Large Relational Databases [HOLO1]:

As stated previously, to record a page of data in a holographic memory we need the two beams, namely, the reference beam and image beam. Once the hologram is recorded, either of the two beams can be used to reconstruct the other beam. A Volume Holographic database System proposed by B.J. Goertzen and P.A. Mitkas is shown in the figure below. It has the following parts – Image beam generation, Reference beam generation, Reference beam detection and Image beam detection. The laser source used is a 50 mW frequency-doubled YAG laser operating at 532 nm. This laser source feeds the reference beam generation and the image beam generation subsystem. Light entering the image beam generation system is modulated by the SLM, which contains the data page to be written in to the memory. This light is filtered and focused on the holographic crystal via lenses L4, L5 and L6. The reference beam generation sub-system generates a unique reference beam for each data page to be written. Moving the stage containing the Lens L1 and Mirror M3 changes the focussed spot in the image plane of the lens L2, thus changing the direction of the reference beam (and hence the address to which the image is written to).

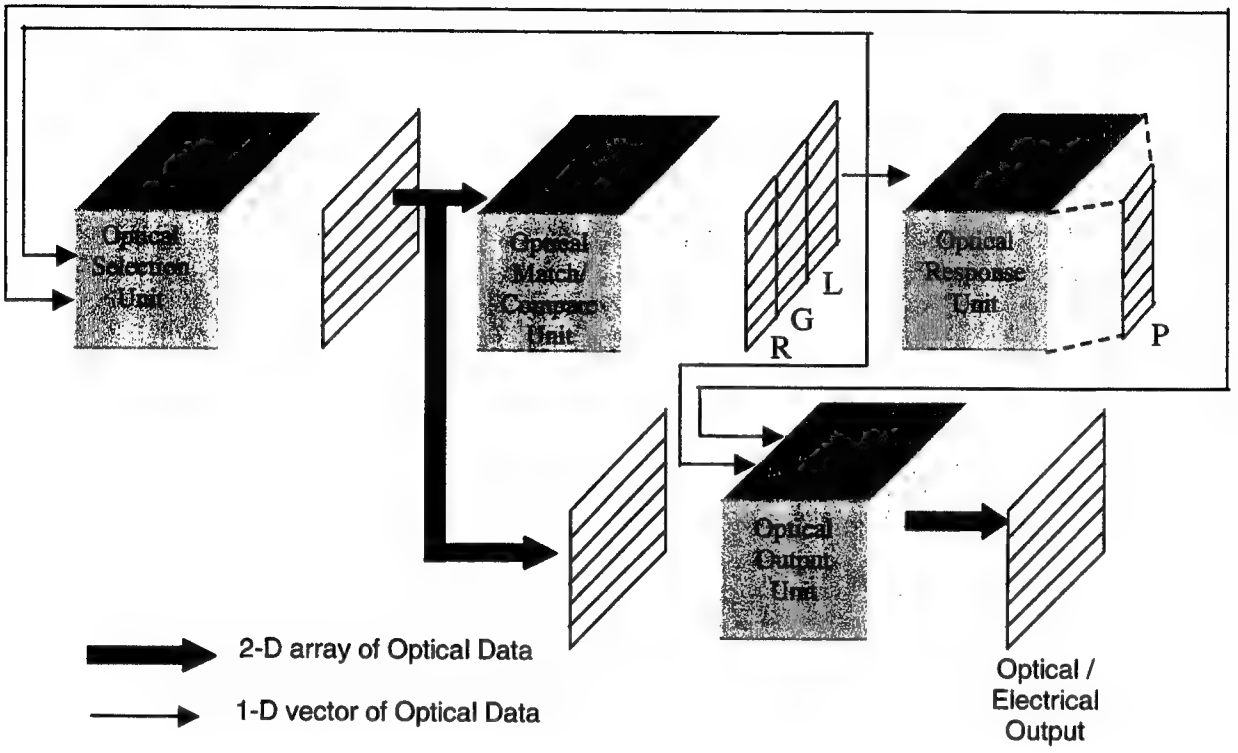
During address based recall, the crystal is illuminated with the reference beam (address). The image beam detection sub-system would now detect the original page that was recorded w.r.t to this reference beam. During associative recall, the crystal is illuminated with the search argument, through the image generation sub-system. Each hologram that correlates with this search argument generates a correlation peak in the correlation plane, corresponding to the reference beam with which it was recorded. These peaks are detected by the reference beam detection sub-system. The authors propose special encoding scheme for the data pages to be stored in the holographic crystal. Please refer [HOLO2] for more details.



Optical System for Volume Holographic Storage System [HOLO2]

7.3.5.2 Optical Content Addressable Parallel Processor [OCAPP1]

The Optical Content Addressable Parallel Processor (OCAPP) proposed by A. Louri, consists of the following units - Optical selection unit, Optical match/compare unit, Optical response unit, Optical output unit. Each of these units are explained briefly below.

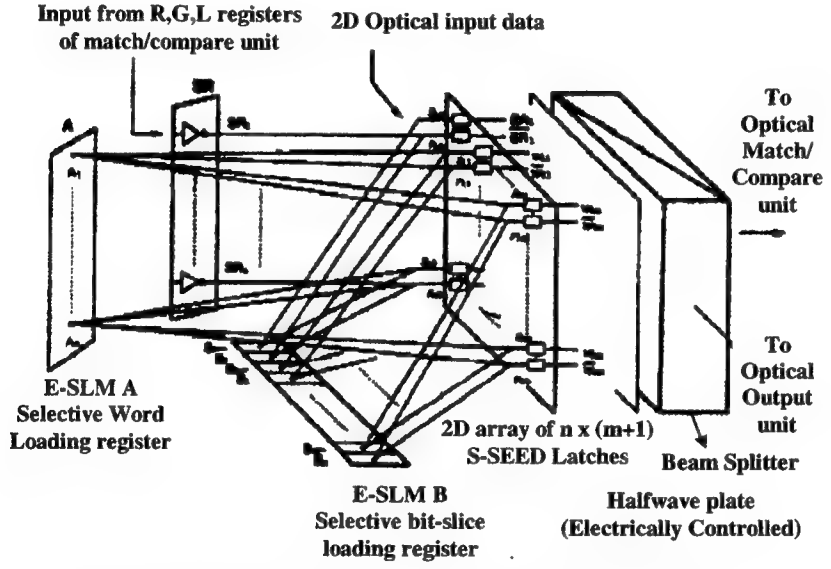


Organization of the Optical Content Addressable Parallel Processor [OCAPP1].

According to the author (A.Louri) the search operations that this processor is capable of performing can be classified as basic search and compound search operations. Basic search operations are simple operations such as equivalence search ($=, \neq$), threshold search ($<, \leq, <, \geq$) and extremum search (MIN, MAX). These operations require no feedback. Compound search operations include sorting, adjacency search, between limits search and outside limits search. These operations are implemented as a series of basic search operations. Please refer [OCAPP1] for algorithms corresponding to these operations. [OCAPP3] has algorithms for relational database operations that could be performed on these OCAPP.

Optical Selection Unit:

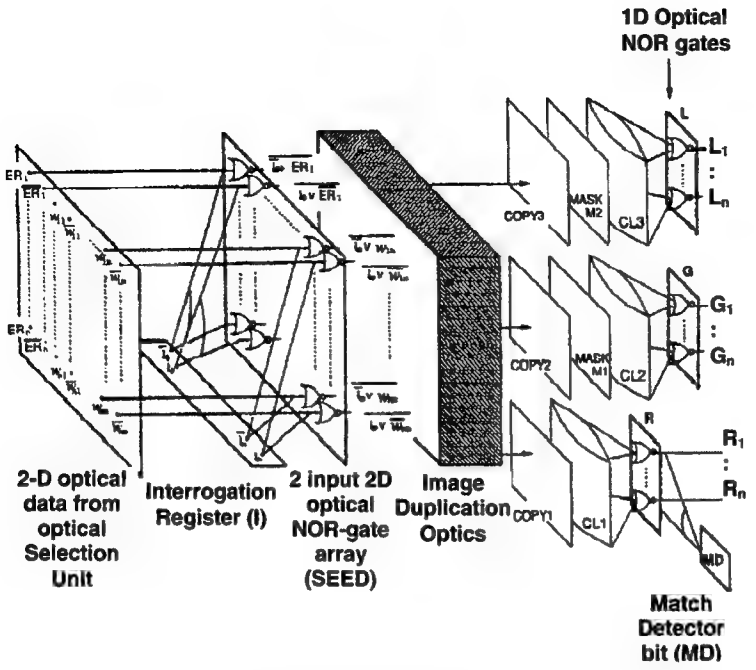
This unit as shown in the following figure, consists of a 'n x (m+1)' (where n – number of words, m – bits/word) storage array, word and bit-slice enable logic. The storage array is an array of clocked S-SEED devices. A clocked 'n x 1' word register A serves to write data words in to the array and a clocked '1 x m' bit slice register B serves to write bit slices into the storage array. Both these registers can be implemented by using E-SLMs so that they can be loaded electrically. The storage array can also be directly loaded from an external optical memory such as Page Oriented Holographic Memory (POHM) or from electronic memory if E-SLM's are used as a transducer. The first column of the storage array is reserved for a 'n x 1' Enable Register (ER). The memory word W_i is enabled if and only if $ER_i = 1$.



Optical implementation of the Selection unit. [OCAPP1]

Optical Match/Compare Unit:

This unit as shown in the figure below, consists of a '1 x m' interrogation register I, logic hardware to perform bitwise comparison between the bits of the interrogation register and the enabled bits in the storage array, three 'n x 1' registers (R, G, L) to hold the results of the comparison. G and L registers can be considered as working registers, and are used for magnitude comparisons (>, <). R - the Response register, is used to hold the result of comparison. A single indicator bit MD (Match Detector) indicates whether or not there are any matching words.



Optical implementation of match compare unit. [OCAPP1]

The bit matching operation is nothing but an XOR operation. Any logic equation can be rewritten in terms of a NOR gates. The equation for the match comparison operation expressed in terms of NOR logic, as derived A. Louri is

$$R_i = \underbrace{\overline{\overline{I_1 + \overline{W_{i1}}} \;\; \overline{I_1 + W_{i1}}}}_{\text{XOR } (I_1, W_{i1})} + \ldots + \underbrace{\overline{\overline{I_m + \overline{W_{im}}} \;\; \overline{I_m + W_{im}}}}_{\text{XOR } (I_m, W_{im})} + \underbrace{\overline{\overline{I_1 + \overline{ER_i}} \;\; \overline{I_1 + ER_i}}}_{\text{Selective Enable/ Disable term}} \qquad \text{---- 1}$$

The selective enable or disable term checks the value of the bit ER_i for each word in the storage array W_i. If $ER_i = 0$, the word W_i is disabled and vice versa. If $R_i = 0$, then $I = W_i$, and we have just found a match.

The relative magnitude comparison (greater and less than operations) is bit serial, and is expressed as

$$G_i = \overline{I_j} + W_{ij} \qquad\qquad L_i = I_j + \overline{W_{ij}}$$

These two terms are actually contained in equation 1. Hence, the optical match/ compare unit can be built as an array of NOR gates, with each NOR gate calculating one of the terms of equation 1, as shown in the figure above. The NOR gate is implemented using the SEED devices

explained earlier. The output of the NOR gates is then replicated in to three copies and necessary optics (cylindrical lenses, mask, 1D NOR gates) is used to fill up the R,G and L registers.

Optical Response unit:

This unit consists of a priority encoder, and a priority register P. The priority encoder allows only the first responder to pass to the register P. The priority circuit can be implemented using several stages of 1D NOR-gate in the form of a binary tree with space-invariant interconnections between them.

Optical Output unit:

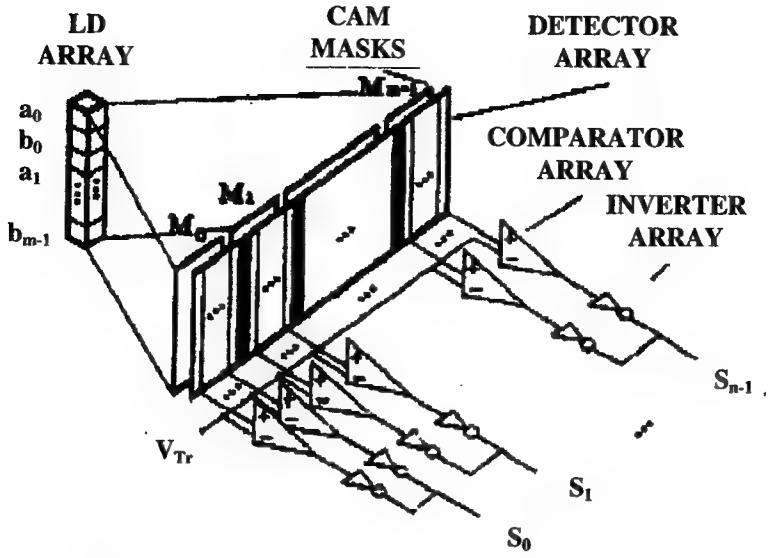
The optical output unit outputs memory words whose corresponding bits in the R, G, L and P registers are asserted.

For a detailed information on this processor and on its performance refer [OCAPP1] and [OCAPP3].

7.3.6 Other Optical Architectures

7.3.6.1 Non-Holographic Content Addressable Memory Based Arithmetic Processor

Most of the above mentioned processors are used for either relational database operations or for image correlation. However, certain optical arithmetic processors have also been proposed. An optical arithmetic processor is usually based on CAMs. An optical CAM can either be implemented using holographic or non-holographic techniques. An optical non-holographic CAM as presented by A. Kostrzewski et al, can be used as an arithmetic processor and is capable of performing single-step binary addition, multiplication. This m-bit CAM processor consists of, two m-bit input numbers $a=a_0a_1...a_{m-1}$ and $b=b_0b_1...b_{m-1}$. Masks M_0, ... M_{n-1} correspond to the CAMs designed for the output bits S_0,..S_{n-1}. L The LD's in the following figure are Laser Diodes and are used to illuminate the CAM Mask. As can been from the figure, the Detector is integrated with the CAM, so as to minimize the longitudinal dimension of the optical system on the mask plane.



Optical Implementation of a non-holographic m-bit CAM processor. [NONHOLO]

The reader is referred to the papers included in the enclosed annotated bibliography for more on these type of non-holographic arithmetic processors.

8 Neural Networks

A neural network does not store, match, and retrieve binary words as most of the previously described associative memories do. It is, nevertheless, an associative memory because it finds the stored information that most closely resembles the key.

A neural network attempts to mimic the human brain, which is composed of many interconnected neurons. Therefore, a neural network is composed of many simple computing nodes interconnected by links. Each link has a numeric weight associated with it. These weights are the primary method of information storage in a neural network.

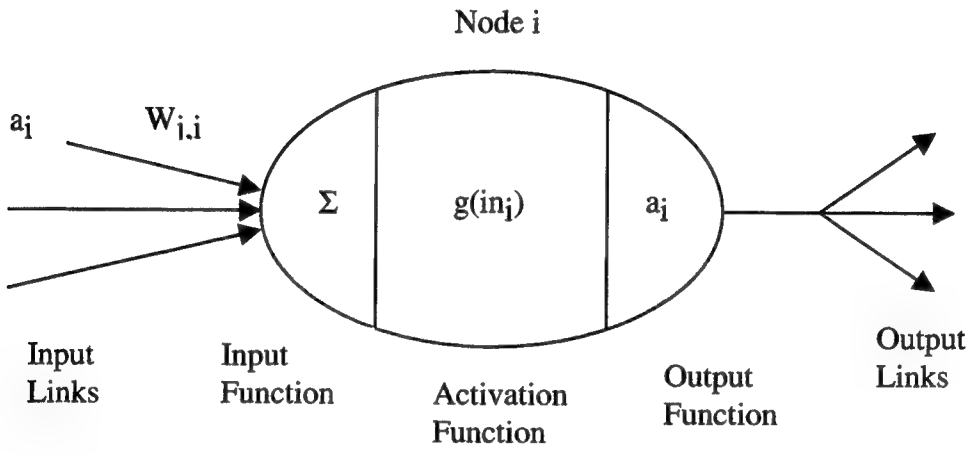
The information is stored by training the neural network, which involves presenting a series of example inputs and the desired output for each. The weights within the network adjust themselves until the desired outputs are obtained.

One of several a specific network topologies, called a Hopfield network, has the behavior that after it is *trained*, it acts as an associative memory. The output will represent the training input that most closely resembles the new input. In this way, the neural network is similar to the optical image correlator described previously.

Section 9.1, below, examines a single node in a neural network and section 7.2 discusses how the nodes can be interconnected.

8.1 A neural network node

As shown below, in the picture from Russell [NEURAL1], a node consists of input links, an input function, an activation function, an output, and output links.



Each input link has a weight, which can be adjusted during training. For example, in the diagram above, the link from node j to node i has weight $W_{j,i}$. The input function computes the total weighted input, which is the sum of the input activations times their respective weights.

$$in_i = \Sigma W_{j,i} \; a_j$$

The non-linear activation function g() is applied to the result computed by the input function. A common choice for the activation function is a step function, where the output is 1 if the input is greater than some threshold and 0 otherwise.

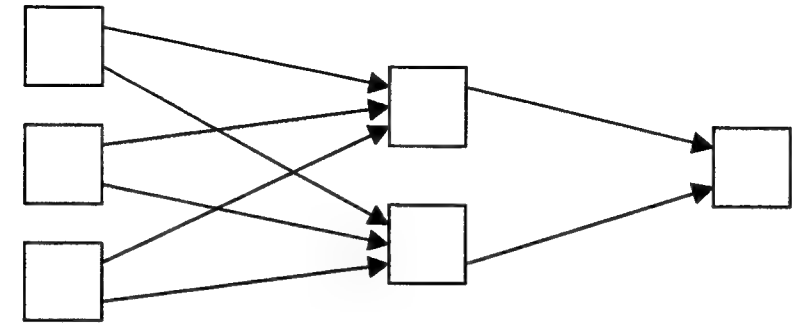
The output function sends the result of the activation function to other nodes.

8.2 Linking nodes together into networks

The nodes described above are linked together into networks that act as associative memories.

The most common structure is a layered feed-forward network, as shown in the figure below. A layer is a group of nodes. A network consists of an input layer, an output layer, and several hidden interior layers. The network is feed-forward because each node is linked only to the next layer; there are no links between units in the same layer, no links backward to a previous layer, and no links that skip a layer. As can be seen, the system computes a weighted sum of the inputs,

and then thresholds the result. This provides a generalization of the hamming distance matching technique shown in the earlier section of this report.



9 Other Common Applications

As previously mentioned, associative memories and processors are well-suited to many common applications, such as databases. This section describes the use of associative memory for databases and several other applications, such as image processing.

9.1 Image Analysis

Associative memories are often used for image analysis because the pixel information can be accessed and processed in parallel.

Snyder[IMAGE3] shows that an associative memory can be used to partition an image into regions. Regions consist of pixels that have similar attributes, such as gray-scale level, and are connected to each other. Each region is assigned a number, which is stored in the associative memory. Once two regions, say regions 1 and 2, have been determined to be equivalent, all occurrences of "2" are found and replaced in parallel by "1".

Shain[IMAGE2] provides another example of using associative memory for image analysis. He and his colleagues demonstrate that an associative memory can be used to perform a discrete cosine transform, which is used in many image compression algorithms. Working one bit-slice at a time, the additions and multiplications required for the discrete cosine transform are performed in a bit-parallel manner.

Storer[IMAGE4] demonstrates that an associative memory can be used for image rendering. Image rendering involves the following tasks: representing objects to be displayed, geometrical transformations on the objects to scale and position them in a universe, viewing transformations to produce coordinates relative to the viewpoint, windowing to determine which objects are

visible or partially visible, combining the objects into an image, and conversion to pixels. Multiple associative processors, ideally one per pixel, are each assigned part of the image to render. The author states that a content addressable parallel processor (CAPP) of 64K processor elements running at 100 MHz can render 44,050 full-color, anti-aliased, depth-buffered, shaded triangles per second. Although this performance is less than that of the best, specialized, graphics processors, the CAPP is more general purpose.

9.2 Vision

Herrman[IMAGE1] describes a vision system where the image is pre-processed in its analog form and then converted to a digital format using an analog-to-digital converter. The digitized image is then loaded into an associative memory, where all further processing is performed. In this system, the associative memory is formed from dynamic electronic cells with 5 transistors each.

The vision system performs image smoothing and segmentation in real time. The image smoothing is done by convolving the image with a 2D kernel as shown below.

	1/8	
1/8	1/2	1/8
	1/8	

In the segmentation step, each pixel is compared with its neighbors and a flag is set if the difference is greater than a given threshold.

The vision system can also be used for stereo matching, which gauges depth information by comparing two images displaced in space. Each associative memory cell contains corresponding pixels from the left and right images. The memory cell compares its two pixels and examines the results of the comparisons in the surrounding cells. Then, one image is shifted relative to the other and the process repeats until the depth can be computed.

More image processing and machine vision algorithms are discussed in Appendix II.

9.3 Database Mining

As previously mentioned, an associative memory is well-suited for database applications. A common database application is data mining, which involves finding patterns in the data. The following example of a data mining problem is from Hall[DATABASE2]. The manager of a grocery story wishes to place things on sale that are commonly bought with other items. For example, potato chips are often bought with dip, so putting potato chips on sale may also increase dip sales. The manager has a database of all the store transactions for the past year, where each record contains all of the items bought by one customer in one visit. He wishes to find groups of items that are often bought together from that database.

This kind of data mining is done by hypothesizing a particular combination of items, searching each record for that combination, and counting the matches. This is then repeated for all other combinations of interest. Each probe in the algorithm is simple, but it requires examining every record of the database.

Hall[DATABASE2] states that a current microprocessor system would be able to examine 1 million records in a few seconds. A conventional-design CAM could do the probe in a few microseconds, but a CAM large enough to hold a million records would be very expensive. The paper then presents an associative architecture, called the Rutgers CAM, that consists of a binary tree of ALUs (arithmetic and logical units) with associative memory units as the leaves. The authors state that the architecture is able to do the probe in a few milliseconds and requires about twice the number of chips as a conventional memory.

10 Summary and Conclusions

This study has shown that associative memory has many advantages over conventional memory. Associative memory can be searched in parallel. Many operations can be performed directly in the memory, so data does not need to be transferred from memory to the CPU.

This study has also shown that the major disadvantages of semiconductor associative memory, which are higher cost, higher power consumption, and lower storage density, can be avoided by the use of optical technology.

Appendix 1: Relational Database Model:

A relational database contains relations, which are often called tables. A relation in the relational database model is defined mathematically as a subset of the Cartesian product of a number of data attributes, where the order of the attributes is arbitrary and can be altered [2PHO2]. A relation is usually represented as a table, where the columns correspond to the data attributes and each row is a tuple, which is often called a record.

A sample table of sports data is shown below:

Number	Name	Position	Batting Avg
3	Jones	Infield	.376
12	Smith	Pitcher	.123
8	Jones	Catcher	.268
7	Doe	Pitcher	.180

Each column is an attribute of the relation, such as "Name" or "Position". Each row is a record, which is a collection of attributes about a single player.

No two rows should be identical in a relation. Every relation should have one or more attributes that serve as unique identifiers of the tuple. These unique identifiers are called primary keys. In the table above, "Number" is the primary key because it is unique – no two players on the same team have the same number. Neither "Name" nor "Position" would be good primary keys because there may be two players with the same name or who play the same position.

A database usually has many relations.

Relational Operations:

1. Projection: Projection operation enables a user to select columns (attributes) from a relation and specify their order. If the primary keys are not used in the projection arguments, then the result might have duplicate entries. In the example below, the Name attribute is projected.

Number	Name	Position	Batting Avg
3	Jones	Infield	.376
12	Smith	Pitcher	.123
8	Jones	Catcher	.268
7	Doe	Pitcher	.180

2. Selection: Selection operation involves selecting of tuples from a relation that satisfy one or more selection criteria. These selected tuples are a result of a comparison between the relations in the database against user-supplied selection arguments. In the example below, the record for player number 8 is selected if the selection criteria is "catcher".

Number	Name	Position	Batting Avg
3	Jones	Infield	.376
12	Smith	Pitcher	.123
8	Jones	Catcher	.268
7	Doe	Pitcher	.180

3. Join: When two relations share a common attribute, they may be joined. The join operation of two operand relations A and B on attributes X (from A) and Y (from B) is the relation C obtained by concatenating each tuple in A with each tuple in B.

Number	Name	ERA
12	Smith	0.89
7	Doe	1.92

Table A

Number	SSN	Expenses
3	111-11-1111	$42.76
12	222-22-2222	$102.95
8	333-33-3333	$3.12
7	444-44-4444	$61.34

Table B

In the example above, information about players 12 and 7 from two separate tables is concatenated together because the player numbers are the same in both tables.

4. Other relational operations include, updating, sorting and range queries. All these operations involve extensive comparing on the input values to some reference which is fixed (Selection) or changing (join, sorting).

Appendix II: Image processing algorithms for a Content Addressable Parallel Processor.

10.1 Processor Architecture

Charles C. Weems in [THESIS1] proposes Content Addressable Array Parallel Processor (CAAPP) for image processing applications. The characteristics of the proposed architecture for the processor contains

- 512 x 512 array of processing elements - since the author is dealing with 512 x 512 pixel images he proposes to have a Processing Element (P.E) per pixel.
- Each P.E has the following single-bit registers – X, Y, Z, A and B where,
  - X – Response bit. It also acts as one of the inputs to the ALU. This register presents its value to the neighboring cells at all times.
  - Y – This register is a backup for the Response (X) register. It also acts as a second input to the ALU.
  - Z – The register holds the carry bit.
  - A – This register contains the activity mask. When this is one, the cell (P.E) is active and responds to all instructions. When zero the cell only responds to a special subset of instructions.
  - B – This register is the backup for the A register.
- All processing inside the P.E. are bit-serial.
- Each P.E has 32 bits of memory. Only the registers inside the P.E can read from and write to the register. Each P.E includes an ALU capable of performing AND, OR, NAND, NOR, NOT, XOR, EQUAL, full add. Since the P.E is bit-serial, these operations are done on individual bits.
- The control unit broadcasts a Comparand (C) to all the P.E's. This is used for the Content Based Search on all the P.E's
- Apart from these, the processor has a global Some/None bit and Response Count Register (RC). The response count provides a count of the response bits (X) that are set. The Some/None bit indicate whether a previous instruction resulted in any response i.e.; if any of the response bit (X) is set.

The definition of a neighbor is crucial in image processing. As previously stated, the X register continually transfers its data to its neighboring cells. The neighbors of a cell are the N, S, E and W neighbors in case of a 4-way neighborhood. An 8-way neighborhood could also be defined in which case the 8 cells surrounding the given cell is considered as its neighborhood. However, the author chooses to use the 4-way neighborhood. The author uses two kinds of connection network between the P.E's. This relates to the fact that author distributes the 512 x 512 P.'s in to 64 circuit boards, each with 64 chips. Each chip in turn have 64 P.E's (512 x 512 P.E = 64 circuit boards x 64 chips/circuit board x 64 P.E/chip). The two kinds of connection network are dead-end connection and zigzag connection. The dead edge connection network treats the edges as if they were a ring of dead cells. In the zigzag connection network,

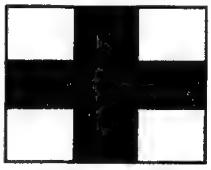
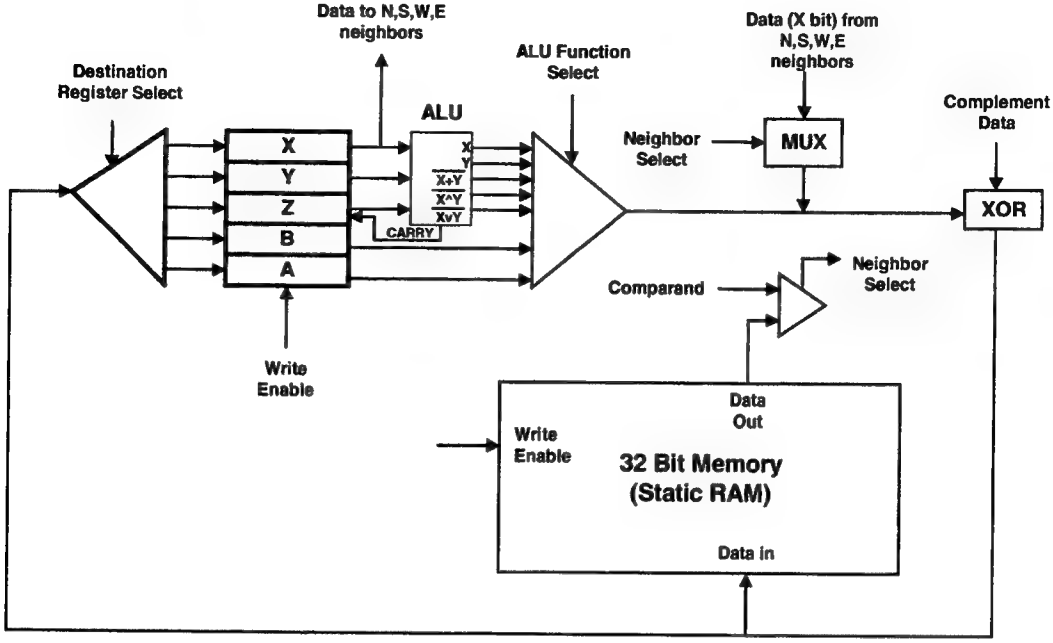
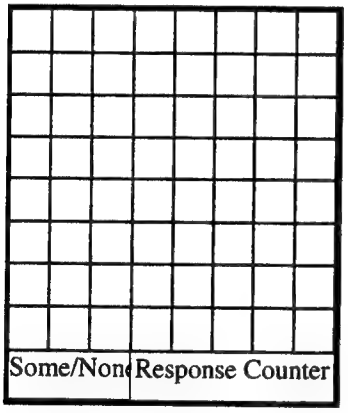


Fig: Neighborhood of a cell

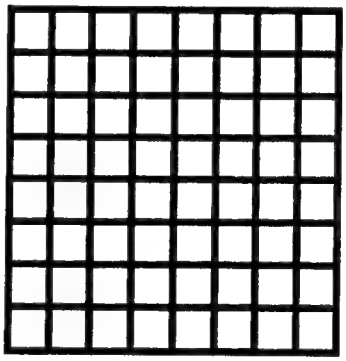
the elements of an edge are connected to the corresponding elements in the opposite edge, shifted over one element.



Organization of one Processing Element



64 P.E's per chip arranged in to 8 x 8 square.



64 chips per circuit board arranged in to 8 x 8 square.

10.2 Algorithms:

The author divides his collection of algorithms in to four parts namely – basic operations, simple image processing operations, higher level image processing operations and other application areas. The basic operations are frequently used instructions that could most likely be called upon by the image processing instructions. The second section (simple image processing instructions) include Guassian Smoothing convolution, Sobel edge detection histrogramming etc. The third section includes complex image processing applications such as region growing and labeling etc. The last section suggests some of the other areas to which CAAPP could be applied.

Notation:

- X, Y, Z, A, B refer to the CAAPP registers.
- M(I) refers to the Ith bit of the Memory cell. (Remember that each P.E has a 32 bit memory).
- C refers to the broadcast comparand.
- Any operation performed by the ALU refers only to the currently active cells (whose A bit is set to one) unless explicitly stated.

10.2.1 Basic Operations:

10.2.1.1 Exact Match

This operation looks for an exact match between certain continuos bits in the Memory (M(S) to M(S+L-1)) and the broadcast Comparand (C), where S is the LSB of the field being matched and L is its length.

Steps:

Note:

M – Memory
S – LSB of the memory bit slice being matched
L – Length of the memory bit slice
C – Broadcast Comparand.

1. Save A in B for all P.E's *// This step saves the activity information // from the previous operation for all cells.*
2. For I = S to S+L-1 do the following for all active cells
   / The following operations are bit serial and are done one bit at a time starting with the LSB */*
   X <= M(I)
   Y <= C
   Z <= 0 *// Carry is zero.*
   / With Z = 0, X+Y is same as X ⊕ Y so that NOT (X + Y) is the equality (NOR) test. Hence if X is equal to Y the result is 1 else the result is zero. */*
   X <= NOT (X + Y)
   / When A is assigned to X, some of the already active cells (A = 1) become inactive since those cells failed the above equality test.*/*
   A <= X
3. Restore A from B for all P.E's. (A <= B)

The result of this operation is available in the X register. When there is a match between the memory bit slice and the Comparand, X is set to 1.

10.2.1.2 Greater than and lesser than Comparand searches.

This operation compares a bit slice from Memory M (M(S) to M(S+L-1)) against the Comparand (C).

Steps:

Note:
M – Memory
S – LSB of the memory bit slice being compared
L – Length of the memory bit slice
C – Broadcast Comparand.

1. Save A in B for all P.E's *// This step saves the activity information from the // previous operation for all cells.*
2. For I = S+L-1 to S do the following for all active cells
   / The following operations are bit serial and are done one bit at a time starting with the **MSB**. */*
   X <= M(I) *// X <= C for 'less than' test*
   Y <= C *// Y <= M(I) for 'less than' test*
   Z <= 0 *// Carry is zero.*

/* NOT (X + Y) is the equality (NOR) test. Hence if X is equal to Y the result is 1 else the result is zero. Suppose if M > C then it means that X = 1 and Y = 0. Since we are doing the 'M > C' test we want the response bit (X) to be set whenever M > C. But in this case X is already a 1. Similarly if M < C (X=0, Y=1) we want the response bit to be set to zero, and in this case it is already a zero. Hence nothing needs to be done. Similar argument hods true for the less than test. Once we know that M is either < or > C we can make this cell inactive.*/

   A <= NOT (X + Y)

3. */* Once you are out of the loop A will be 1 only for those cells whose M matched with C. Since we are only interested in 'M > C' searches, we change the response bit for all the active cells to 0. */*
   Set X <= 0 for all the active cells
4. Restore A from B for all P.E's. (A <= B)

The result of this operation is available in the X register.

A. Greatest and least searches.

This operation selects all cells from among the currently active cells which have highest (or lowest) from among the active cells in the array.

Steps for the greatest number search:

Note
M – Memory
S – LSB of the memory bit slice being matched
L – Length of the memory bit slice
C – Broadcast Comparand.

1. Save A in B for all P.E's
2. For I = S+L-1 to S do the following for all active cells
   / The following operations are bit serial and are done one bit at a time starting with the **MSB**. */*
   X <= M(I).
   / When X bit of any of the P.E is one, the Some/None bit is set to 1. In that case, we know that the greatest number from among the active cells, should have a 1 in the I-th MSB position. Hence we can disable those cells that do not have a 1 in the I-th position. If the Some/None bit is not set, then we know that the greatest number has a zero in the I-th bit position and so we proceed to the next bit. */*
   If Some then */* i.e.; if Some/None bit is set */*
   A <= X
3. Restore A from B for all P.E's. (A <= B)

Steps for the least number search:

Note:
M – Memory
S – LSB of the memory bit slice being matched
L – Length of the memory bit slice
C – Broadcast Comparand.

1. Save A in B for all P.E's
2. For I = S+L-1 to S do the following for all active cells
   / The following operations are bit serial and are done one bit at a time starting with the **MSB**. */*
   X <= M(I).
   / When X bit of any of the P.E is one, the Some/None bit is set to 1. In that case, we know that the least number from among the active cells, should have a 0 in the I-th MSB position. Hence we can disable those cells that do not have a 0 in the I-th position. If the Some/None bit is not set, then we know that the least number has a zero in the I-th bit position and so we proceed to the next bit. */*
   If Some then */* i.e.; if Some/None bit is set */*
   A <= NOT(X)
3. Restore A from B for all P.E's. (A <= B)

10.2.1.3 Select first search:

This operation is used when a CAAPP operation results in multiple responders. This operations selects only one responder out of the multiple responders, with activity turned off in the remainder of them. This algorithm also returns the <X,Y> location of the selected cell to the controller. Higher priority is given to the responder that is topmost and leftmost.

Steps:

/ The following steps save the original response patterns in Z and activity patterns in B and prepares the response store for the search. */*

1. Turn on all chip column selects
2. Turn on all chip row selects.
3. If None then exit.
4. Save A in B for all P.E's (B <= A)

5. Set A for all cells. (A <= 1)
6. Z <= X (Save X in Z)
7. Clear responders for inactive cells. (A <= NOT(B), X <= 0)
8. Save the Masked responders in Y (A <= 1 for all cells, Y <= X).
9. *Perform a binary search on the rows of the chips in the array to find the topmost with atleast one responder. This is done by the external controller which selectively turns on certain rows (of chips) and reads the Some/None bit. At the end of this process only that row of chip is selected.*

/ The following steps uses one cell per column activity mask and the on-chip shift network to find the topmost row with an active responder (within a chip). */*

10. I <= 1 *// Loop Variable*
11. X <= 0 *// Sets the responders of all active cells to zero*
12. Turn on the responders for the top row of cells only.
13. A <= X *// Only the Topmost row in a chip is now active.*
14. X <= Y *// Restore the Masked responders*
15. While None do the following

/* If none then we know that there is no responder in this row. Hence we turn off the activity of this row and proceed to the next row */

X <= 1 *//Turns on the responders for the currently active cells*
Shift the responders by one row.
A <= X for all cells *// Current row is not active anymore. The next row is active*
X <= Y *// Restore the Masked Responders*
I <= I + 1 *// Increment loop variable.*

/ At the end of the above loop the loop variable I has the row number of the topmost row that had an active responder. Also at the end of the loop this topmost row will be active for all chips. */*

16. *Perform a binary search on the columns of the chips in the array to find the leftmost with atleast one responder. This is done by the external controller which selectively turns on certain columns (of chips) and reads the Some/None bit. At the end of this operation, the leftmost chip with a responder in the topmost row of chips is selected.*

/ The following steps which column of cells within the chip contains the leftmost responder. */*

17. J <= 1 *// Loop Variable*
18. X <= 0 *// Sets the responders of all active cells to zero*
19. Turn on the responders for the leftmost column of cells only.

/ Note that by the virtue of step 15, only the topmost row (with an active responder) is now active. All other rows are inactive. Hence step 19 basically turns on the leftmost cell in that active row */*

20. A <= NOT(X)
21. X <= 0 *// Clears all of the other X bits, so that now all cells have X = 0, except our cell of interest (that we selected in step 19)*
22. A <= NOT(A) *// That cell(from step 19) is the only active cell*
23. X <= Y *// Restore the Masked responders*
24. While None do the following

/* If none then we know that there is no responder in this cell. Hence we turn off the activity of this cell and proceed to the next cell (move right) */

X <= 1 *//Turns on the responders for the currently active cells*
Move one cell to the right.
A <= X for all cells *// Current cell is not active anymore. The next cell to the right is active*

X <= Y *// Restore the Masked Responders*
J <= J + 1 *// Increment loop variable.*

/ At the end of the above loop the loop variable J has the column number of the topmost-leftmost column that had an active responder. */*

A combination of (I, J) gives the coordinates of the topmost-leftmost active cell. But it does not indicate which chip (out of the possible 64 chips) has this responder. That information is obtained from steps 9 and 16.

10.2.1.4 Add and Subtract Constant

The broadcast comparand C (also called as Constant) is added to or subtracted from the memory bit slice (M(S) to M(S+L-1)), where S – LSB of the field and L – Length of the field. The result is stored back in the memory in the same location (M(S) to M(S+L-1)),
and the carry (overflow in addition and underflow in subtraction) is stored at M(S+L).

Steps:

Note

M – Memory
S – LSB of the memory bit slice being matched
L – Length of the memory bit slice
C – Broadcast Comparand.

1. Z <= 0 // Carry is zero for addition, and 1 for subtraction.
2. For I = 0 to L-1 do the following
   X <= M(S + I)
   Y <= C(I) // Y <= NOT(C(I)) for subtraction
   X <= X + Y
   M(S + I) <= X
3. X <= Z *// Store Carry in X. This is an indication of overflow / underflow. Since X is the response bit, further processing can be done based on the value in X*
4. M(S + L) <= X *// The carry is also stored at this location in Memory.*

10.2.1.5 Add and Subtract fields

In this operation 2 memory bit slices ($M(S_1)$ to $M(S_1+L)$ and $M(S_2)$ to $M(S_2+L)$) are added (subtracted) to each other. The result is stored back in the memory in the same location as the first operand ($M(S_1)$ to $M(S_1+L-1)$), and the carry (overflow in addition and underflow in subtraction) is stored at $M(S_1+L)$.

Steps:

1. Z <= 0 // Carry is zero for addition, and 1 for subtraction.
2. For I = 0 to L-1 do the following
   X <= M(S_1 + I)
   Y <= M(S_2 + I)
   Y <= NOT(Y) // This step is required only for subtraction. Note that since memory output cannot be inverted (and only register outputs can be inverted we could not use Y <=NOT(M(S_2 + I)) in the previous step.)
   X <= X + Y
   M(S_1 + I) <= X
3. X <= Z *// Store Carry in X. This is an indication of overflow / underflow. Since X is the response bit, further processing can be done based on the value in X*
4. M(S_1 + L) <= X *// The carry is also stored at this location in Memory.*

10.2.1.6 Multiply by Constant

This operation uses the 'Shift and add' method to perform the multiplication. The Memory bit slice M(SM) to M(SM + L-1) are multiplied with the broadcast comparand and the result is stored in the memory at (M(SR) to M(SR+LC+LM-1), where LM is the length of the multiplicand and LC is the length of the constant.

Steps:

```
/* Clear the result field */
for I=0 to LC+LM-1 do the following
        M (SR + I) <= 0
for I=0 to LC –1 do the following
 /* Check for C(I). If it's a zero, we need to do nothing. If it's a 1 we must shift the Multiplicand I times and
                                                    add it to the result */
        if C(I) = 1 then
                Z <= 0  // Carry is zero for addition.
                for J = 0 to LM –1 do
                        X <= M(SM + J)
                        Y <= M(SR + J + I)
                        Y <= X + Y  // Multiplicand is shifted by I bits and added to the result.
                        M(SR + J + I) <= Y
                // end of the J loop
                X <= Z // Store Carry in X
                M(SR + LM + I) <= X          // Store carry in the memory (so that it will be added the next
                                             // time we do a shift and add .
        // end of if
// end of the I loop.
```

10.2.1.7 Divide by Constant

In this method the Memory bit slice M(SD) to M(ED) is divided by the broadcast comparand and the result is stored in the memory at (M(SQ) to M(EQ), where SD is the LSB of the dividend,

ED is the MSB of the dividend, SQ is the LSB of the quotient and EQ is the MSB of the quotient. Three of the previously defined are used here, namely, Compare greater than equal, Subtract constant, Add constant. LF is the length of the fields.

Steps:

1. Clear Quotient Memory
2. if (Constant = 0) or (Constant >= 2**(LF-1)) then **exit.**

// The following steps align the divisor

3. K <= 0 // Loop variable
4. while C(LF) <> 1 *// <> means not equal to*
   Shift C left
   K <= K + 1
5. B <= A for all cells *// Save the activity pattern*

// Set Activity for those cells whose Dividend (Memory bit slice) is >= Divisor

6. for I = K to 0 do
   for J=LF-1 to 0 do
   X <= M(SD + J)
   Y <= C(J)
   Z <= 0
   A <= - (X + Y) *// '>=' test*
7. X <= 1 *// Response bit is 1 for all active cells*
8. A <= B for all cells *// Restore activity*
9. A <= X *// Only those cells that were initially active, and whose X = 1 (so* that *// Dividend >= Divisor) are now active*

// Now we are ready to subtract the Divisor from the dividend for all the active cells.

10. Z <= 1 *// Carry is 1 for division.*
11. for J = 0 to LF –1 do
    X <= M(SD + J)
    Y <= NOT (C(J))
    X <= X + Y *// Subtraction*
    M(SD + J) <= X
12. X <= Z // Store final carry in X
13. M (ED + J) <= X

// Now that we did the subtraction, the next step is to add 2^I to the quotient.

14. Z <= 1
15. Y <= 0
16. for J = SQ + I to EQ do *// Note the starting value of J.*
    X <= M(J)
    X <= X + Y *// Z = 1 and Y = 0 is equivalent to increment by 1*
    M(J) <= X
17. X <= Z
18. M(EQ +1) <= X
19. Shift C right
20. Restore Activity for all cells (A <= B)

10.2.1.8 Multiply Fields

This operation uses the 'Shift and add' method to perform the multiplication. The Memory bit slice M(SM) to M(SM + LM-1) are multiplied with the another memory bit slice M(SF) to M(SF+LF-1) and the result is stored in the memory at (M(SR) to M(SR+LM+LF-1), where LM is the length of the multiplier and LF is the length of the multiplicand. SM, SF and SR are the LSB of the multiplicand, multiplier and the result respectively.

Steps:

Save the Activity pattern in B (B <= A).
/ Clear the result field */*
for I=0 to LM+LF-1 do the following
    M (SR + I) <= 0
for I=0 to LM –1 do the following
/* Check for C(I). If it's a zero, we need to do nothing. If it's a 1 we must shift the Multiplicand I times and add it to the result */
    A <= M(SM + I) *// Activate cells with a 1 at the I th bit in the multiplicand, so that we do // the shift and add only in those cells.*
    Z <= 0 *// Carry is zero for addition.*
    for J = 0 to LF –1 do
        X <= M(SF + J)
        Y <= M(SR + J + I)
        Y <= X + Y *// Multiplicand is shifted by I bits and added to the result.*
        M(SR + J + I) <= Y
    // end of the J loop
    X <= Z *// Store Carry in X*
    M(SR + LM + I) <= X *// Store carry in the memory (so that it will be added the next // time we do a shift and add .*
    A <= B for all cells *// Restore the activity so that we can proceed with the next shift // and add*
// end of the I loop.

10.2.1.9 Divide Fields

This operation is similar to the Divide by Constant operation except that the divisor here is also a memory bit slice rather than the broadcast comparand. Please refer [Thesis1, pages 223-228] for more details.

10.2.1.10 Response Count

This operation returns a count of the number of elements which have their X registers set to one. Note that every chip has its own response count register.

Steps:

1. A <= 1
2. Clear Response Count Register

/ The following is for a single chip */*

3. for I = 1 to 64 do
   Shift and Count responders

/ The following is for a circuit board that has 64 such chips */*

4. Turn off all chip row selects.
5. Turn on all chip column select lines
6. for I = 1 to 64 do
   Turn on row select line I
   Pipeline Add North to South
7. for I = 1 to 8 do
   Pipeline Add North to South
8. for I = 1 to 64 do
   Pipeline Add West to East.

10.2.2 Simple Image Processing Applications

10.2.2.1 Conway's Game of life

Problem definition:

The basic idea of the game is to represent the growth culture of cells given certain simple rules about how cells are born and how they die. The cells live in square grid, with one cell per grid. The eight cells surrounding this cell are its neighborhood. A given cell either lives or dies depending on the number of neighbors it has. If a live cell has 2 or 3 live neighbors it lives to the next generation. If an empty grid position has exactly 3 neighbors, a cell will be born in this grid in the next generation. Hence the algorithm determines the next generation by simply counting the number of live cells in the neighborhood. Appropriate tests are made on this count to determine the contents of the cell in the next generation.

Pseudo code:

This algorithm uses the bit zero of memory to hold the status of the cell (Live or dead) and bits 1 through 4 to hold the count of live neighbor cells.

XNW	XN	XNE
XW	XM	XE
XSW	XS	XSE

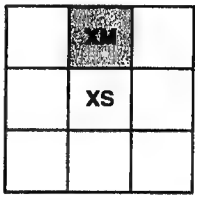
Note: In the diagram, X_M – is the X bit of the main cell M.
X_N – is the X bit of the North neighbor of M
X_S – is the X bit of the South neighbor of M
and so on.

Steps:

1. A <= 1 for all cells *// Activate all cells.*
2. for N = 1 to 4 do
   M(N) <= 0 *// Clear counter*
3. X <= M(0) *// Load Status in to X. Recall that only the X*
   // register can be read by 4 neighbors (N,S,E,W).

/ The following two steps sends the current X to North. Hence if M is the current cell, then M receives X from S and passes on its X to N. */*

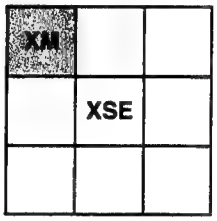
4. Shift_X_North.
5. M(1) <= X



Array after Shift_X_North

/ Shift to NorthWest. */*

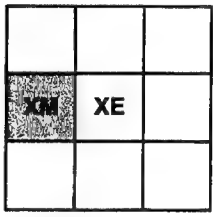
6. Shift_X_West
7. If X=1 increment counter M(4 to 1)



Array after Shift_X_West

/ Shift to West.*/*

8. Shift_X_South
9. If X=1 increment counter M(4 to 1)



Array after Shift_X_South

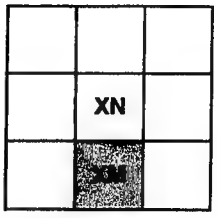
/ Shift to Southwest.*/*

10. Shift_X_South
11. If X=1 increment counter M(4 to 1)

Array after Shift_X_South

/ Shift to South.*/*

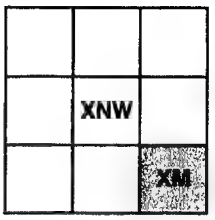
12. Shift_X_East
13. If X=1 increment counter M(4 to 1)



Array after Shift_X_East

/ Shift to Southeast.*/*

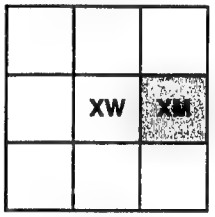
14. Shift_X_East
15. If X=1 increment counter M(4 to 1)



Array after Shift_X_East

/ Shift to East.*/*

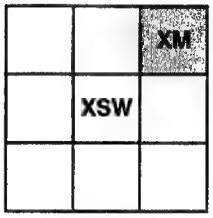
16. Shift_X_North
17. If X=1 increment counter M(4 to 1)



Array after Shift_X_North

/ Shift to Northeast.*/*

18. Shift_X_North
19. If X=1 increment counter M(4 to 1)



Array after Shift_X_North

/ Evaluate the count */*

20. Evaluate the count and set the status bit (M(0)) accordingly.

10.2.2.2 Guassian Smoothing image Convolution

A discrete 2-D convolution is based on a mask of multipliers that each cell applies to its local neighborhood, forming the sum of the pair-wise products of cell's neighbors with their corresponding mask values.

Thus in the fig, the new value of V_0 after convolution (with the mask) is

$V_0' = \Sigma V_i . M_i$, i = 0 to 8.
where, Vi – is the value inside the cell i.

M_1	M_2	M_3
M_8	M_0	M_4
M_7	M_6	M_5

MASK

V_1	V_2	V_3
V_8	V_0	V_4
V_7	V_6	V_5

V_0 (central cell) and its neighborhood

This update is performed after all the cells have finished examining their neighborhoods.

The function mask for Guassian convolution is

1	2	1
2	4	2
1	2	1

The following algorithm is similar to the algorithm for 'Conway's game of life in the sense that in this algorithm, instead of X_m, the cells value (Vi) goes around its neighborhood. Now instead of incrementing the counter (if X = 1) we multiply V_i with the corresponding mask value M_i.

Pseudo code

Steps.

Note:

SV – LSB of X_0
SS – LSB of Sum (X_0')
ST – LSB of temporary storage.
ES – MSB of Sum (X_0')
LF – Length of temporary storage.

```
// Multiply V0 by 4.
1. for I = 0 to LF –1 do
     X <= M(SV + 1)
     M (SS + I + 2) <= X   // Shift and add by 3, since 4 is 100 in binary
2.   for I = SS + LF + 1 to ES do
          M(I) <= 0        // Initialize the rest of the sum bits

// Send V0 to North: Load V0 bit-serially in to X and send it to the north neighbor.
3.  Z <= 0                 // Carry is zero for addition
4.  For I = 0 to LF –1 do
          X <= M (SV + I)  // Get the next bit from V0 starting with LSB
          Shift _X_North   // Send it to the cell above. Note that at the same time the
                           // current cell will receive a value from its South cell.
          M (ST + I) <= X  // Store this in the cells temporary register
          Y <= M (SS + I + 1)  // Shift sum by 2 (10 in binary).
          Y <= X + Y       // Add
          M(SS + I + 1) <= Y   // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
```

```
5.  X <= 0
6.  For I <= SS + LF + 1 to ES do
            Y <= M(I)
            Y <= X + Y
            M(I) <= Y
```

// Send V_0 to Northwest

```
7.  Z <= 0                      // Carry is zero for addition
8.  For I = 0 to LF –1 do
            X <= M (SV + I)     // Get the next bit from V0 starting with LSB
            Shift _X_West       // Send it to the cell above. Note that at the same time the
                                // current cell will receive a value from its South cell.
            M (ST + I) <= X     // Store this in the cells temporary register
            Y <= M (SS + I )    // No shift. Multiply by 1.
            Y <= X + Y          // Add
            M(SS + I + 1) <= Y  // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
9.  X <= 0
10. For I <= SS + LF + 1 to ES do
            Y <= M(I)
            Y <= X + Y
            M(I) <= Y
```

// Send V_0 to West

```
11. Z <= 0                      // Carry is zero for addition
12. For I = 0 to LF –1 do
            X <= M (SV + I)     // Get the next bit from V0 starting with LSB
            Shift _X_South      // Send it to the cell above. Note that at the same time the
                                // current cell will receive a value from its South cell.
            M (ST + I) <= X     // Store this in the cells temporary register
            Y <= M (SS + I + 1 ) // Shift sum by 2 (10 in binary).
            Y <= X + Y          // Add
            M(SS + I + 1) <= Y  // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
13. X <= 0
14. For I <= SS + LF + 1 to ES do
            Y <= M(I)
            Y <= X + Y
            M(I) <= Y
```

// Send V_0 to Southwest

```
15. Z <= 0                      // Carry is zero for addition
16. For I = 0 to LF –1 do
            X <= M (SV + I)     // Get the next bit from V0 starting with LSB
            Shift _X_South      // Send it to the cell above. Note that at the same time the
```

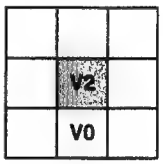
```
                                    // current cell will receive a value from its South cell.
        M (ST + I) <= X             // Store this in the cells temporary register
        Y <= M (SS + I )            // No shift. Multiply by 1.
        Y <= X + Y                  // Add
        M(SS + I + 1) <= Y          // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
17. X <= 0
18. For I <= SS + LF + 1 to ES do
        Y <= M(I)
        Y <= X + Y
        M(I) <= Y


// Send V0 to South
19. Z <= 0                          // Carry is zero for addition
20. For I = 0 to LF –1 do
        X <= M (SV + I)             // Get the next bit from V0 starting with LSB
        Shift _X_East// Send it to the cell above. Note that at the same time the
                                    // current cell will receive a value from its South cell.
        M (ST + I) <= X             // Store this in the cells temporary register
        Y <= M (SS + I + 1 )        // Shift sum by 2 (10 in binary).
        Y <= X + Y                  // Add
        M(SS + I + 1) <= Y          // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
21. X <= 0
22. For I <= SS + LF + 1 to ES do
        Y <= M(I)
        Y <= X + Y
        M(I) <= Y

// Send V0 to Southeast
23. Z <= 0                          // Carry is zero for addition
24. For I = 0 to LF –1 do
        X <= M (SV + I)             // Get the next bit from V0 starting with LSB
        Shift _X_East// Send it to the cell above. Note that at the same time the
                                    // current cell will receive a value from its South cell.
        M (ST + I) <= X             // Store this in the cells temporary register
        Y <= M (SS + I )            // No shift. Multiply by 1.
        Y <= X + Y                  // Add
        M(SS + I + 1) <= Y          // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
25. X <= 0
26. For I <= SS + LF + 1 to ES do
        Y <= M(I)
        Y <= X + Y
        M(I) <= Y
```

```
// Send V0 to East
27. Z <= 0                          // Carry is zero for addition
28. For I = 0 to LF –1 do
        X <= M (SV + I)             // Get the next bit from V0 starting with LSB
        Shift _X_North              // Send it to the cell above. Note that at the same time the
                                    // current cell will receive a value from its South cell.
        M (ST + I) <= X             // Store this in the cells temporary register
        Y <= M (SS + I + 1 )        // Shift sum by 2 (10 in binary).
        Y <= X + Y                  // Add
        M(SS + I + 1) <= Y          // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
29. X <= 0
30. For I <= SS + LF + 1 to ES do
        Y <= M(I)
        Y <= X + Y
        M(I) <= Y
```

```
// Send V0 to Northeast
31. Z <= 0                          // Carry is zero for addition
32. For I = 0 to LF –1 do
        X <= M (SV + I)             // Get the next bit from V0 starting with LSB
        Shift _X_North              // Send it to the cell above. Note that at the same time the
                                    // current cell will receive a value from its South cell.
        M (ST + I) <= X             // Store this in the cells temporary register
        Y <= M (SS + I )            // No shift. Multiply by 1.
        Y <= X + Y                  // Add
        M(SS + I + 1) <= Y          // Store the sum back
// Propagate the carry (if any) to the higher order bits of sum
33. X <= 0
34. For I <= SS + LF + 1 to ES do
        Y <= M(I)
        Y <= X + Y
        M(I) <= Y
    // Normalize
    for I = 0 to LF –1 do
        X <= M(SS + I + 4)          // Divide by 16.
        M (SV + I) <= X
```



10.2.2.3 Sobel Edge detection

A common operation in image processing and computer vision systems is the reduction of an image to a set of lines that represent the edges of the image. This can be achieved using the Sobel

operator. This operation has two different masks that are applied to the image to get the horizontal and vertical gradient of the image.

-1	0	1
-2	0	2
-1	0	1

F(x) - Horizontal Gradient Mask

1	2	1
0	0	0
-1	-2	-1

F(y) - Vertical Gradient Mask

V1	V2	V3
V8	V0	V4
V7	V6	V5

Neighborhood of the cell V0. Vi is the value stored in the cell i.

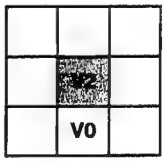
Pseudo code:

Note:

M(XR) to M(XR + LF) – Memory bit slice where X gradient result is stored.
XR – LSB of X-gradient result
M(YR) to M(YR + LF) – Memory bit slice where Y gradient result is stored.
YR – LSB of Y-gradient result
LF – Length of these fields.

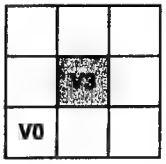
As you will see, the shifts employed in the algorithm is a distribution process, rather than a collection process. Hence the fumction mask must be mirrored across the central cell. For eg, when the cells value is stored in the north neighbor, the south mask multiplier must be applied. Although the same method was use din the previous algorithm it did not cause a problem because the mask used in that algorithm was symmetric.

1. Load V_0 in to X and send it to the current cells south neighbor. Store this is in the temporary register in the receiving cell. **Note, that since multiplication is done in the receiving cell, the received value corresponds to the north neighbor of the receiving cell.** Hence **the** must me mirrored across the central cell. **actually** the mask

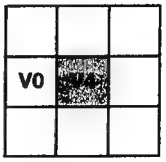


2. Multiply the received X by 2 and store the result in M(YR) to M(YR + LF)

3. Load the value in the temporary register in to X and send it to the neighbor of the current cell. Store this value in the temporary in the receiving cell. west register

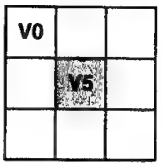


4. Store the received X in M(XR) to M(XR + LF) (multiply by 1). the received X with the Y result in M(YR) to M(YR + LF) and result back in M(YR) to M(YR + LF). Also add store the



5. Load the value in the temporary register in to X and send it to the north neighbor of the current cell. Store this value in the temporary register in the receiving cell.
6. Multiply the received X by 2 and add it to the X result in M(XR) to M(XR + LF) and store the result back in M(XR) to M(XR + LF).

7. Load the value in the temporary register in to X and send it to the neighbor of the current cell. Store this value in the temporary in the receiving cell.
8. Multiply the received X by 1, add it to the X result in M(XR) to LF) and store the result back in M(XR) to M(XR + LF).
9. Multiply the received X by -1, add it to the Y result in M(YR) to LF) and store the result back in M(YR) to M(YR + LF).

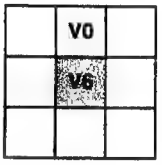


north register

M(XR +

M(YR +

10. Load the value in the temporary register in to X and send it to the neighbor of the current cell. Store this value in the temporary in the receiving cell.
11. Multiply the received X by -2 and add it to the Y result in M(YR) to LF) and store the result back in M(YR) to M(YR + LF).



east register

M(YR +

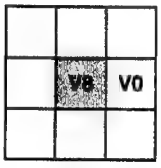
12. Load the value in the temporary register in to X and send it to the neighbor of the current cell. Store this value in the temporary in the receiving cell.
13. Multiply the received X by –1, add it to the X result in M(XR) to LF) and store the result back in M(XR) to M(XR + LF).
14. Multiply the received X by –1, add it to the Y result in M(YR) to LF) and store the result back in M(YR) to M(YR + LF).

east register

M(XR +

M(YR +

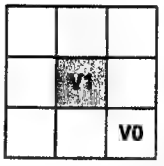
15. Load the value in the temporary register in to X and send it to the neighbor of the current cell. Store this value in the temporary in the receiving cell.
16. Multiply the received X by -2 and add it to the X result in M(XR) to LF) and store the result back in M(XR) to M(XR + LF).



south register

M(XR +

17. Load the value in the temporary register in to X and send it to the neighbor of the current cell. Store this value in the temporary in the receiving cell.
18. Multiply the received X by –1, add it to the X result in M(XR) to LF) and store the result back in M(XR) to M(XR + LF).
19. Multiply the received X by 1, add it to the Y result in M(YR) to M(YR + LF) and store the result back in M(YR) to M(YR + LF).



east register

M(XR +

10.2.2.3.1 Histogram

A histogram is a graph that represents the frequency with which a local image event or image feature occurs, and peaks in the histogram indicate those values that occur most often.

Valleys in histogram indicate values that occur infrequently and are thus likely to be found on edges or transitions.

The following algorithm uses the above mentioned "Select Less Than" algorithm and "Count Responders" algorithm. The algorithm selects a range of values starting from the lowest range and working up to the highest range. A particular range of values is called bucket

Steps:

Note:

Max – is the greatest value in any cell
Size – number of values in the range assigned to each bucket
High – holds the current maximum values for the bucket being filled.

1. Y <= 1
2. Size <= Max div Buckets
3. for I = 1 to Buckets do
   High <= I * Size
   Select cells < High
   X <= X AND Y
   Bucket[I] <= Response count
   X <= NOT(X) *// Turn off the response bit for the currently active cells.*
   Y = X AND Y *// Y is zero for the currently active cells, so that they will be*
   // considered further inside the loop

10.2.3 Higher Level image Processing Applications

10.2.3.1 Region growing and Labeling.:

This is a basic operation in image processing applications wherein adjacent pixels are given the same label (number) if they are similar, or have no edges between them. Adjacent pixels are nothing but the neighbors of the pixel. A pixel can either have 4 neighbors (N, S, W, E) or 8 neighbors (N, NE, E, SE, S, SW, W, NW). The following pseudo code uses the 4-way neighborhood.

In the following pseudo code the status bit holds a one if a cell is part of an edge and zero otherwise. The initial label for a cell is an 18 bit concatenation of its X and Y coordinates. Recall that since we have 512 x 512 cells the X and Y coordinates each have 9 bits.

Pseudo code:

Copy status bit from each neighbor into own memory
Load X-Y grid in to cells
Repeat

```
Clear 'Label-Changed' bit
For each neighbor (N,S,E,W) do
        If Neighbor-Status <> 0 // if the neighbor is not a boundary
                If Own-Label > Neighbor's-Label and Own-status is not boundary
                        Copy Neighbor's-Label in to Own-Label
                        Set Label-Changed bit
```

Until no label changed bits are set.

10.2.3.2 Other operations

Other operations that could be done using CAAPP are Rotation of 3D model and extraction of frontal surface, Decomposition of rotational and translational motion parameters from optic flow, center of mass, Geocorrection of satellite images, Square gird sort, Real time execution of Lisp programs, Simulation of neural networks. Please refer [THESIS1] for a detailed psuedo code for all of the above mentioned operations.

Annotated Bibliography

Part I: Items Referenced in this Report

General associative memory and processors

[GEN1] L. Chisvin and R. J. Duckworth, **Content-Addressable and Associative Memory: Alternatives to the Ubiquitous RAM**, Computer, vol. 22, pp 51-64, July 1989.

Abstract: Researchers have understood the basic principles of storing and retrieving data by content rather than by address for about 30 years. Despite this relatively long incubation period, information has spread slowly from the academic arena, and the technology has not been available to produce a successful commercial product. As a result, many designers have not developed the skills to work with associative and content-addressable memories. However, VLSI technology has improved the feasibility of associative systems and overcome many implementation obstacles.

Our goal in this article is to explain content-addressable and associative processing systems to people unfamiliar with them and show that such systems are now commercially feasible. We first contrast content-based archiving and retrieving with address-based storage methods. We then explain the principles, advantages, and obstacles of content-addressable storage, describe associative processing methods and architectures, and finally, discuss software for associative systems.

[GEN2] C. C. Foster, Content Addressable Parallel Processors, Van Nostrand Reinhold, New York, 1976.

Abstract: This book discusses semiconductor associative memory and contains quite a few algorithms to take advantage of them. It discusses applications for content addressable parallel processors, most of which take advantage of a parallel search. The book also explains the STARAN computer developed by Goodyear Aerospace, which was a commercially available computer with associative memory.

[GEN3] K. E. Grosspietsch, **Associative Processors and Memories: A Survey**, IEEE Micro, vol. 12, pp 12-19, June 1992.

Abstract: Because of declining hardware prices, associative (content-addressable) architectures are again gaining importance, especially in artificial intelligence and database applications. This survey introduces the classical content-addressable memory and explains its realization at the transistor level. Then it describes some unorthodox CAM approaches, discusses associative processor systems, and classifies current approaches.

[GEN4] L. J. Irakliotis, G. A. Betzos, and P.A. Mitkas, **Optical Associative Processing**, in Associative Processors and Processing, A. Krikelis and C. Wheems, editors, IEEE Computer Society Press, Los Alamitos California, 1997.

Abstract: Optical and optoelectronic schemes for associative processing are presented and reviewed. We discuss both analog and digital associative processing techniques with an emphasis on non-numerical processing applications such as image processing and relational database operations. A brief review of optical storage systems and other necessary optical components precedes the discussion of system implementation.

[GEN5] R. M. Lea, **ASP: A Cost-effective Parallel Microcomputer**, IEEE Micro, Vol. 8 No. 5, pp 10-29, October 1988.

Abstract: Associative String Processor microcomputers provide highly versatile components for the low-cost implementation of high-performance information processing systems. By mapping application data structures to a string representation and supporting content addressing and parallel processing, ASP achieves both application flexibility and a step-function improvement in cost-performance figures. This improvement occurs without the loss of computational efficiency usually suffered by general-purpose parallel processors.

[GEN6] S. P. Levitan and D. M. Chiarulli, **Proposal for Associative Memory Study: Architectures and Technology**, University of Pittsburgh Office of Grants and Contracts, 1999.

Abstract: This study will analyze the technologies and architectures currently available to support associative (content addressable) memory systems for both large (data base) and high speed applications. The results of the study will be a report summarizing likely avenues of research with highest payoff potential.

[GEN7] J. L. Potter, **Associative Computing Environment**, in Associative Processors and Processing, A. Krikelis and C. Wheems, editors, IEEE Computer Society Press, Los Alamitos California, 1997.

This paper points out the need for improved software productivity. It shows how associative computing with a tabular data structure and natural language syntax can be combined to provide an associative computing environment (ACE) suitable for introducing data parallelism to elementary algebra students and bypassing the Basic-Pascal-C intermediate stages. This ACE approach also provides considerable promise for improved productivity.

[GEN8] D. Sima, T. Fountain, and P. Kacsuk, Advanced Computer Architectures: A Design Space Approach, Addison Wesley, Harlow England, 1997, Chapter 12, pp 455-483.

Abstract: This chapter introduces the basic ideas underpinning associative and neural architectures. It describes how these ideas are translated into practice in associative processing, using the Associative String Processor as an example, and briefly discusses the ease of mapping applications onto an associative system. It also discusses the comparatively new field of neural computers.

Optical content-addressable parallel processors and devices

[OCAPP1]: Louri-A, **Optical content-addressable parallel processor: architecture, Algorithms, and design concepts**, *Applied-Optics*, vol.31, no.17; 10 June 1992; p.3241-58.

Abstract: Associative processing based on content-addressable memories has been argued to be the natural solution for nonnumerical information processing applications. Unfortunately, the implementation requirements of these architectures when one uses conventional electronic technology have been cost prohibitive; therefore associative processors have not been realized. Instead, software methods that emulate the behavior of associative processing have been promoted and mapped onto conventional location-addressable systems. However, this does not bring about the natural parallelism of associative processing, namely, the ability to access many data words simultaneously. Optics has the advantage over electronics of directly supporting associative processing by providing economic and efficient interconnects, massive parallelism, and high-speed processing. The principles of designing an optical content-addressable parallel processor (OCAPP) for the efficient support of parallel symbolic computing are presented. The architecture is designed to exploit optics advantages fully in interconnects and high-speed operations. Several parallel search-and-retrieval algorithms are mapped onto an OCAPP to illustrate its capability of supporting parallel symbolic computing. A theoretical performance analysis of these algorithms is presented. This analysis reveals that the execution times of the parallel algorithms presented are independent of the problem size, which makes the OCAPP suitable for applications in which the number of data sets to be operated on is high (e.g. massive parallel processing). A preliminary optical implementation of the architecture with currently available optical components is also presented.

[OCAPP2] Peng-Yin-Choo; Detofsky-A; Louri-A , **Multiwavelength optical content-addressable parallel processor for high-speed parallel relational database processing,** Applied-Optics. vol.38, no.26; 10 Sept. 1999; p.5594-604.

Abstract: We present a novel, to our knowledge, architecture for parallel database processing called the multiwavelength optical content-addressable parallel processor (MW-OCAPP). The MW-OCAPP is designed to provide efficient parallel data retrieval and processing by means of moving the bulk of database operations from electronics to optics. It combines a parallel model of computation with the many-degrees-of-processing freedom that light provides. The MW-OCAPP uses a polarization and wavelength-encoding scheme to achieve a high level of parallelism. Distinctive features of the proposed architecture include (1) the use of a multiwavelength encoding scheme to enhance processing parallelism, (2) multicomparand word-parallel bit-parallel equality and magnitude comparison with an execution time independent of the data size or the word size, (3) the implementation of a suite of 11 database primitives, and (4) multicomparand two-dimensional data processing. The MW-OCAPP architecture realizes 11 relational database primitives: difference, intersection, union, conditional selection, maximum, minimum, join, product, projection, division, and update. Most of these operations execute in constant time, independent of the data size. We outline the architectural concepts and motivation

behind the MW-OCAPP's design and describe the architecture required for implementing the equality and intersection-difference processing cores. Additionally, a physical demonstration of the multiwavelength equality operation is presented, and a performance analysis of the proposed system is provided.

[OCAPP3] James A. Hatch Jr, **The Design, Development and Demostration of an Optical Content-Addressable Parallel Processor for High-Speed Database Processing**, Master of Science Thesis, University of Arizona, 1994

Abstract: This thesis extend the concept of Optical Content Addressable Parallel Processor to a novel architecture designed specifically for the parallel and high speed implementation of database operations called Optical Content-Addressable Parallel Processor for Relational database Processing (OCAPPRP). OCAPPRP combines a parallel model of computation – associative processing with a parallel and high speed technology optics. This report presents an overview of the architecture, followed by its optical implementation. Algorithms for the full set of relational operations are presented to illustrate the architecture's potential for efficiently supporting high speed database processing.

[OECMOS] A. H. Sayles and J. P. Uyemura, **An optoelectronic CMOS memory circuit for parallel detection and storage of optical data**, *IEEE Journal of Solid State Circuits*, Vol.26, p 1110, 1991. (ref in 2PHO2)

Abstract: A CMOS static RAM (SRAM) circuit capable of detecting and storing optically transmitted data is described. Bits of data are transferred to the memory circuit via an array of parallel light beams. A 16-b optoelectronic SRAM was fabricated in a standard bulk CMOS process and tested using argon and helium-neon lasers. Data contained in an array of 16 light beams with an average power of 3.35 mu W/pixel were successfully transferred to the SRAM in parallel fashion. The storage of the optical information was verified by electronically addressing each cell. The optical data transfer technology is extended to other systems in which high speed and parallelism are essential.

Database applications using associative processors

[DATABASE1] S. Akyokus and P. B. Berra, **Optical Content Addressable Memories for Data/Knowledge Base Processing**, Proceedings of the Fifth International Parallel Processing Symposium, Anaheim CA, 1991.

Abstract: This paper describes the principles and architectures of a one and two-dimensional optical content addressable memory (OCAM). These architectures are based on optical matrix multipliers which use free space interconnections in optics. They work as a CAM due to spatial coding used to perform bit matching and masking. A 1-D OCAM can perform a single (one-to-many) search operation and a 2-D OCAM can perform multiple (many-to-many) search operations in one step. An optical database machine (ODM) based on a 2-D OCAM along with a page-oriented holographic memory (POHM) is proposed for the execution of relational database operations.

[DATABASE2] J. S. Hall, D. E. Smith, and S. Y. Levy, **Database Mining and Matching in the Rutgers CAM**, in Associative Processors and Processing, A. Krikelis and C. Wheems, editors, IEEE Computer Society Press, Los Alamitos California, 1997.

Abstract: The Rutgers CAM architecture is an implementation of memory based on the set of operations that can be done with bit-serial, word-parallel algorithms on a classical content addressable memory. The design reduces the number of active elements, but increases speed and storage capacity compared with the original form. An expanded set of collective functions over the classic model drastically improves its flexibility in data representation, as well as increasing the number of parallel algorithms the architecture can execute.

Database mining and mass matching are applications that require performing simple operations on every element of large databases. Associative processors in general, and the Rutgers CAM in particular, are very efficient on tasks of this kind.

Image processing and vision applications using associative processors

[IMAGE1] F. P. Herman and C. G. Sodini, **A Dynamic Associative Processor for Machine Vision Applications**, IEEE Micro, Vol. 12 No. 3, pp 31-41, June 1992.

Abstract: Massively parallel associative processors may be well suited as coprocessors for accelerating machine vision applications. They achieve very fine granularity, as every word of memory functions as a simple processing element. A dense, dynamic, content-addressable memory cell supports fully parallel operation, and pitch-matched word logic improves arithmetic performance with minimal area cost. As asynchronous reconfigurable mesh network handles interprocessor communication and image input/output, and an area-efficient pass-transistor circuit counts and prioritizes responders

[IMAGE2] Y. Shain, A. Akerib, R. Adar, **Associative Architecture for Fast DCT**, Proceedings of the 1998 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 5, pp 3109-3112, 1998.

Abstract: This paper discusses an associative processor architecture designed to meet the demands of real-time image processing applications. In a single chip, this architecture provides thousands of processors – one for each pixel, in the form of associative memory. This paper focuses on a generic, proprietary associative processor architecture and discusses implementing the discrete cosine transform (DCT) using processors based on this architecture.

Processors based on our associative architecture can process the large amounts of data typically required in real-time imaging applications at a lower cost-performance ratio than conventional processors. The scalable nature of memory-based processor architecture allows developers to rapidly increase processing power without altering the fundamental processor, or system architecture.

[IMAGE3] W.E. Snyder and C.A. Savage, **Content-Addressable Read/Write Memories for Image Analysis**, IEEE Transactions on Computers, vol. C-31, No. 10, pp 963-968, October 1982.

Two common problems in image analysis are described – the region-labeling problem and the clustering problem. Both are shown to be instances of a search-and-rename problem which can be solved in parallel by a system architecture inherently suitable for VLSI implementation. That architecture, a novel type of content-addressable memory, is described, and its application to search-and-rename problems is discussed.

[IMAGE4] R. Storer, **Image Rendering with Content-Addressable Parallel Processors**, in Associative Processors and Processing, A. Krikelis and C. Wheems, editors, IEEE Computer Society Press, Los Alamitos California, 1997.

Abstract: Associative processors based on content-addressable memory (CAM) often have a simple CPU attached to each memory word. The associative behavior of the CAM is used to select a subset of processors that perform the next associative or arithmetic operation.

As the operation can be applied to all memory words simultaneously, these processors can be classed as SIMD parallel processors. Like other SIMD machines, associative processors are most efficient when used for processing large quantities of data, each of which requires an identical sequence of operations.

As many computer graphics and image-processing operations have this characteristic, an associative processor can be used as an intelligent frame store, where each pixel is processed by its own processing element.

Existing, favored computer graphics algorithms tend not to work well when recoded for SIMD implementation because the criteria for optimizing an algorithm for a sequential, or MIMD, machine are different from those for a SIMD machine.

In this paper, I examine the requirements for suitable SIMD algorithms for image rendering in an intelligent, associative frame store. SIMD algorithms are described for producing anti-aliased circles, lines, and two- and three-dimensional polygons.

I describe a practical scheme for implementing these algorithms on an array of associative rendering engines and outline performance predictions for this scheme, using the GLiTCH associative processor.

Neural networks

[NEURAL1] S. J. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, Englewood Cliffs NJ, 1995, Chapter 19, pp 563-597.

From a computational viewpoint, this chapter is about a method of representing functions using networks of simple arithmetic computing elements, and about methods for learning such

representations from examples. From a biological viewpoint, these networks are a mathematical model for the operation of the brain. These networks represent functions in much the same way that circuits consisting of simple logic gates represent Boolean functions. Such representations are particularly useful for complex functions with continuous-valued outputs and large numbers of noisy inputs, where logic-based techniques sometimes have difficulty.

Self-electrooptic effect devices (SEED)

[SEED1] Lentine-AL, Tooley-FAP, Walker-SL, McCormick-FB-Jr, Morrison-RL, Chirovsky-LMF, Focht-MW, Freund-JM, Guth-GD, Leibenguth-RE, Przybylek-GJ, Smith-LE, D'Asaro-LA, Miller-DAB, **Logic self-electrooptic effect devices: quantum-well optoelectronic multiport logic gates, multiplexers, demultiplexers, and shift registers**, *IEEE-Journal-of-Quantum-Electronics*. vol.28, no.6, June 1992, p.1539-53.

Abstract: Two-dimensional arrays of logic self-electrooptic effect devices (L-SEEDs), consisting of electrically connected quantum-well p-i-n diode detectors and modulators are demonstrated. The topology of the electrical connections between the detectors is equivalent to the connections between transistors in CMOS circuits. Three different L-SEED arrays were built and tested. Each element in one array can implement any of the four basic Boolean logic functions (i.e., NOR, NAND, AND, OR). Each element in the second L-SEED array can implement the function E=AB+CD. The third L-SEED array consists of 32*16 arrays of symmetric SEEDs (S-SEEDs) connected with optoelectronic transmission gates. Photonic switching nodes, multiplexers, demultiplexers, and shift registers have been demonstrated using this array. [INSPEC].

[SEED2] A. L. Lentine, D. A. B. Miller, **Evolution of the SEED technology: bistable logic gates to optoelectronic smart pixels**, *IEEE J. of Quantum Electronics*, **29**, 655-669 (1993).

Abstract: The recent evolution of quantum-well self-electrooptic effect devices (SEEDs) for application in free-space optical switching and computing systems is reviewed. Requirements of these systems have stimulated the development of devices usable in large systems of cascaded devices (the symmetric SEED), large two-dimensional arrays of these devices with improved physical performance, logically smarter extensions of these devices (logic-SEEDs), and devices integrating electronic transistors with quantum-well modulators and detectors for both reducing the required optical energies and increasing functionality. This progress and its implications for future developments are summarized. [INSPEC].

[SEED3] H.S. Hinton and A.L.Lentine, **Multiple Quantum-Well Technology takes SEED,** *IEE Circuits and Devices*, p - 12-18, March 1993

Abstract: Progress in the development of self-electrooptic-effect devices (SEEDs) is discussed. The devices include the resistor-SEED (R-SEED) device, which can be viewed as a simple NOR gate. The symmetric SEED (S-SEED) and the logic-SEED (L-SEED) devices with improved features, functionality, and performance are also considered. The integration of FETs with multiple quantum well (MQW) modulators (FET-SEED), enables optical interconnections of

electronic circuits. Where the SEED technology can be used is discussed, and an experimental optical switching fabric made using these devices is described. [INSPEC]

Spatial Light Modulators (SLM)

[SLM1] Neff-JA; Athale-RA; Lee-SH , **Two-dimensional spatial light modulators: a tutorial,** *Proceedings-of-the-IEEE,* vol.78, no.5; May 1990; p.826-55.

Abstract: Two-dimensional spatial light modulators (SLMs) modulate one of the properties of an optical wavefront (amplitude, phase, polarization) as a function of two spatial dimensions and time in response to information-bearing control signals that may be either optical or electrical. These devices form a critical part of optical information processing systems, serving as input transducers as well as performing several basic processing operations on optical wavefronts. A tutorial overview of the 2-D SLMs is given. their applications are outlined, a classification scheme for them is given, and major types of SLMs that are under active development are described. [INSPEC].

[SLM2] D. Flannery, A. Biernacki, J.Loomis and S. Cartwright, " Real time Coherent Correlator using binary magneto-optic spatial light modulators at input and Fourier planes", *Applied-Optics,* vol.25, no.4; 15 Feb. 1986; p.466.

Abstract: An experimental correlator using the Light-Mod magnetooptic spatial light modulator (SLM) devices for real-time modulation of both input images and Fourier plane filters has been constructed and demonstrated. Results are in good agreement with computer simulations based on the FFT. The experiment, results and comparison with theory are briefly described.

Optical Disks

[DISK1] Psaltis D, Neifeld MA, Yamamura A, Kobayashi S , **Optical Memory Disks In Optical Information-Processing,** *Applied Optics,* Vol 29: (14) 2038-2057 MAY 10 1990

Abstract: The authors describe the use of **optical memory disks** as elements in optical information processing architectures. The optical disk is an optical memory device with a storage capacity approaching 10/sup 10/ bits which is naturally suited to parallel access. They discuss optical disk characteristics which are important in optical computing systems such as contrast, diffraction efficiency, and phase uniformity. They describe techniques for holographic storage on optical disks and present reconstructions of several types of computer-generated holograms. Various optical information processing architectures are described for applications such as database retrieval, neural network implementation, and image correlation. Selected systems are experimentally demonstrated. [INSPEC]

[DISK2] Neifeld MA, Psaltis D, **Programmable Image Associative Memory Using An Optical Disk And A Photorefractive Crystal,** *Applied Optics* 32: (23) 4398-4409 AUG 10 1993

Abstract: The optical disk is a computer-addressable binary storage medium with very high capacity. More than 1010 bits of information can be recorded on a 12-cm-diameter optical disk. The natural two-dimensional format of the data recorded on an optical disk makes this medium

particularly attractive for the storage of images and holograms, while parallel access provides a convenient mechanism through which such data may be retrieved. In this paper we discuss a closed-loop optical associative memory based on the optical disk. This system incorporates image correlation, using photorefractive media to compute the best association in a shift-invariant fashion. When presented with a partial or noisy version of one of the images stored on the opticaldisk, the optical system evolves to a stable state in which those stored images that best match the input are temporally locked in the loop.

[DISK3] Mitkas-PA; Berra-PB, **PHOEBUS: an optoelectronic database machine based on parallel optical disks,** *Journal-of-Parallel-and-Distributed-Computing* vol.17, no.3; March 1993; p.230-44.

Abstract: Presents a hybrid optoelectronic relational database machine, called PHOEBUS, that combines optical bulk storage and processing along with electronic control, fast memory, and postprocessing capabilities. The relational database is stored on parallel optical disks in a way that allows the concurrent retrieval of an entire tuple. The data are appropriately arranged on the disk surface and the rotation of the disk is used as the scanning mechanism. High transfer rates from these disks along with parallel execution of relational operations ensure short response times for a large variety of database transactions. [INSPEC]

Two photon memory

[2PHO1] A.S Dvornikov, I.Cokgar, F.McCormick, R.Piyaket, S.Esener, P.M. Rentzepis, **Molecular Transformation as a means for 3D optical memory device**, *Optics Communications* 128, pp. 205-210, 15 July 1996

Abstract: 3D optical memory devices, capable of huge storage, very large bandwidth and amiable to processing, are described. The 3D storage and accessing of information is based on non-linear absorption and emission by two different molecular structures.

[2PHO2] Pericles A. Mitkas and Leo J. Irakliotis, **Three-dimensional optical storage for database processing,** invited paper, *Optical Memory and Neural Networks,* Volume 3, Number 2, pp. 217, 1994.

Abstract: The continuous expansion in volume of current and future databases dictates the development of massive secondary devices that allow parallel access and exhibit high data transfer rates. Efficient storage and retrieval schemes must be developed to replace sequential exhaustive search with intelligent more selective searches. Optical three-dimensional memories, such as volume holograms and two-photon memories, can satisfy the requirements for high volumetric capacity and parallel access. In this paper, we present two architectures for database storage and processing that take advantage of the unique characteristics of these two types of optical memories. Through custom-designed data encoding schemes and parallel optoelectronic processing in the form of data filtering, these systems can reduce the amount of data per transaction that must be accessed and transferred to the electronic front-end.

[2PHO3] S. Hunter, F. Kiamilev, S. Esener, D. A. Parthenopoulos, and P. M. Rentzepis, **Potentials of Two-photon Three-dimensional Optical Memories for High Performance Computing**, *Applied Optics*, Vol. 29, No. 14, pp 2058-2066, May 1990.

Abstract: The advent of optoelectronic computers and highly parallel electronic processors has brought about a need for storage systems with enormous memory capacity and memory bandwidth. These demands cannot be met with current memory technologies (i.e., semiconductor, magnetic, or optical disk) without having the memory system completely dominate the processors in terms of the overall cost, power consumption, volume, and weight. As a solution, the authors propose an optical volume memory based on the two-photon effect which allows for high density and parallel access. In addition, the two-photon 3D memory system has the advantages of having high capacity and throughput which may overcome the disadvantages of current memories. [INSPEC]

[2PHO4] Alexander S. Dvornikov, Ilkan Cokgar, Mark Wang, Fredrick B. McCormick, Jr, Sadik C. Esener, Peter M. Rentzepis, **Materials and Systems for Two Photon 3-D Rom Devices,** *IEEE Transactions on Components, Packaging and Manufacturing Technology – Part A*, Vol 20, No.2 June 1997.

Abstract: The methods and systems used for storing and accessing information in three dimensions by means of two-photon absorption are described. The materials into which the information is stored are organic molecules dispersed in polymer matrices, which change structure and spectra after absorption of light. The writing and accessing of the information can be performed either bit-by-bit or in a two-dimensional (2-D) multibit plane format. Automated recording and readout three-dimensional (3-D) systems have been constructed and characterized. Channel error sources have been identified, and a custom spatial bit-error-rate test have been developed.

[2PHO5] Pericles A Mitkas, Leo J. Irakliotis, Fred R. Beyette Jr., Stuart A. Feld, and Carl W. Wilmsen, **Optoelectronic Data Filter for Selection and Projection,** *Applied Optics*, Vol. 33, N0. 8, pp 1345 -1353, 10 March 1994.

Abstract: Database processing, like the majority of nonnumerical applications, exhibits a high degree of functional parallelism but does not require complex operations therefore it is amenable to optical solutions. The architecture of an optoelectronic filter that is capable of performing selection and projection operations on a two-dimensional data array in a relational database environment is presented. The system receives input from a parallel optical memory, one page at a time, and performs logic operations by using optoelectronic smart pixels based on heterostructure phototransistors and vertical-cavity surface-emitting lasers. Combinations of AND and XOR gates are used to realize row-column masking and comparisons of input data against user-supplied search arguments. The main goal of the filter is to reduce the effective data rate between the highly parallel optical storage and the low input data rate conventional electronic computer, thus efficiently interfacing currently available photonic and electronic technologies. [INSPEC]

[2PHO6] Olson-BH; Paturi-R; Esener-SC, **Biorthogonally accessed three-dimensional two-photon memory for relational database operations,** *Applied-Optics,* vol.36, no.17; 10 June 1997; p.3877-88.

Abstract: Memory bandwidth is a bottleneck for very large database machines. Parallel-access three-dimensional two-photon memories have the potential of achieving enormous throughput (>100 Gbit/s) and capacity (1 Tbit/cm/sup 3/) [Appl. Opt. 29, 2058 (1990)] and, consequently, are well suited for this application. Our analysis shows that some operations can be completed more than 2 orders of magnitude faster with this type of memory than with a system based on serial-access storage. These particular memories have a further feature of being accessible in orthogonal directions. We show that this property, used in conjunction with a three-dimensional data-organization scheme designed for this approach, leads to improved performance by permitting the user a choice of accessing strategies for a given operation.

Holographic memory

[HOLO1] Demetri Psaltis and Fai Mok, **Holographic Memories**, *Scientific American*, Vol23, No 55, pp. 70-76, Nov 1995

No Abstract.

[HOLO2] Burr-GW; Kobras-S; Hanssen-H; Coufal-H, **Content-addressable data storage by use of volume holograms,** *Applied Optics,* vol.38, no.32; 10 Nov. 1999; p.6779-84.

Abstract: Data stored as volume holograms-optical interference patterns imprinted into a photosensitive storage material-can be accessed both by address and by content. An optical correlation-based search compares each input query against all stored records simultaneously, a massively parallel but inherently noisy analog process. With data encoding and signal post-processing we demonstrate a holographic content-addressable data-storage system that searches digital data with high search fidelity. [HOLO2].

Non-holographic memory

[NONHOLO] Kostrzewski-A; Yao-Li; Dai-Hyun-Kim; Eichmann-G, **Non-holographic content addressable memory based arithmetic processor,** *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering,* vol.1230; 1990; p.725-8.

Abstract: A new scheme for digital optical computing, utilizing a non-holographic opto-electronic content addressable memory (CAM), is discussed. To illustrate the performance of a CAM based arithmetic processor, an optical binary carry look-ahead adder (CLA), a binary and a residue multiplier were designed. Experimental results are also presented.

Part II: Items Not Referenced in this Report

General

A. Krikelis and C. Weems, editors, Associative Processors and Processing, IEEE Computer Society Press, Los Alamitos California, 1997.

This book appears at a time when associative processing re-emerges as one of the most interesting topics in computer science, both from theoretical and practical viewpoints. Existing computer models, especially supercomputing and parallel processing architectures, are too expensive and , sometimes, too narrow for the ever expanding user requirements. There is increasing evidence that the associative processing paradigm may be a better and more cost-effective alternative for a variety of applications. In addition, the nature of associative programming constitutes a natural bridge between the classical von Neumann computer architectures and newly emerging neural and genetic computational approaches. Although associative processor designs were initially proposed over thirty years ago, their semiconductor implementation was considered to be too costly in comparison to traditional memory and logic devices. However, progress in microelectronics has greatly reduced the implementation cost of practical associative systems, and they can now be viewed as an inexpensive alternative to complex and costly parallel processing arrays.

This book presents a collection of original papers together with selected papers from recent issues of *Computer* and *IEEE Micro* magazines. It is a cross-disciplinary presentation of the fundamentals, architectures, hardware, software, and applications of associative processing. This book also discusses the problems and solutions of practical implementations of associative processing and processors, highlights strengths and weaknesses of associative processing models and architectural paradigms, and addresses associative processing software and applications.

Associative and Content Addressable Memory

Baum,-Eric-B, **Building an associative memory vastly larger than the brain,** *Science*, 1995 Apr; Vol 268(5210): 583-585.

Discusses the possibility of using tools of molecular biology to produce an associative, or content addressable, memory of immense capabilities. Content addressable memories are useful in many computer contexts and are thought to be an important component of human intelligence. In the simplest approach, 2 DNA subsequences would be assigned to each component of code. Once representatives of component subsequences have been created, they could be cheaply copied. This technology could also be used as an ordinary RAM. With present technology, search operations using beads would be performed on single-strand DNA, while sequencing and restriction mapping would be done on double-strand DNA. DNA-based computing could conceivably produce a technological basis for superhuman intelligence.

Ghose-K , **The architecture of response-pipelined content addressable memories,** *Microprocessing-&-Microprogramming*, vol.40, no.6; July 1994; p.387-410.

Content addressable memories (CAMs) are useful as accelerators for search-intensive programs such as rule-based systems, database applications, documents retrieval and pattern matching. Many applications in these areas required searching along words that are too wide to fit into most commercially available CAM chips. To date, only one commercial chip is available with on-chip support for searching multi-word keys associatively. This chip, however, needs K cycles to search logical keys that are K words long. In this paper, we introduce the architecture of a CAM chip that allows cascading to increase the word-size and the number of words, and yet maintain the search rate constant through the use of a technique called response pipelining. Response pipelining allows the logical word size to be increased by a factor of K but maintains a search and operation completion rate of one per cycle, i.e. the search rate remains independent of K. Response pipelining provides this ability by incorporating very negligible hardware in each chip; response pipelining also allows the board level wiring to be kept very simple. Prototype CAM chips (Response Pipelined CAMs-RPCAMS) have been implemented in a 2-micron CMOS process with 2 metal layers. We describe the response pipelining technique and its potential advantages, as well as the architectural details of the RPCAM chips. Results of using the RPCAM as accelerators for search-intensive applications that require relatively long keys show that significant performance gains are possible through their use. The RPCAM chips are thus ideal as building blocks for large **associative** arrays.

Hurson-AR; Pakzad-S, **Modular scheme for designing special purpose associative memories and beyond,** *VLSI-Design,*. vol.2, no.3; 1994; p.267-86.

The use of associative memories-storage devices that allow data retrieval based on contents-has often been suggested to speed up the performance of many applications. Until recently, using such content-addressable memories (CAMs) was unfeasible due to their high hardware cost. However, the advent of VLSI has made the class of fully-parallel associative memory cost-effective for implementation. This paper briefly overviews the design of several fully parallel associative memories proposed in the literature, concentrating on the design of fully-parallel theta -search CAMs. Existing market realities require that product development be fast and predictable. As a result, design flexibility and automation are becoming increasingly important design features. Using the various CAM designs reviewed, the paper collects the features of these designs into a general, modular CAM organization and describes its major components. The modular CAM organization can be used to design application specific CAMs of varying degrees of functionality. Design and space complexity of a sample associative memory suitable for relational database operations is studied. Finally, the application of genetic algorithms as a means of developing automated design tools for the fabrication of modular VLSI design chips is discussed. Given a library of CAM modules, the desired functionality and a set of speed and area constraints, this optimization technique produces a suitable CAM design. The proposed technique has been implemented and its performance measure is briefly addressed.

Jalaleddine-SM , **Associative memories and processors: the exact match paradigm**, *Journal-of-King-Saud-University-(Computer-and-Information-Sciences).* vol.11; 1999; p.45-66.

Associative or content addressable memories (CAM) are crucial in the implementation of high performance computing architectures for applications that require intensive data management or are cognitive in nature. The basic architecture of associative memories can be based on either the exact match or neural network models. This paper focuses on exact match associative memories. The milestone achievements in the field since the first associative memory implementation four decades ago are discussed. A classification of the diverse associative computing architectures is presented. It comprises of two levels of distinction, the associative memory organization and the processing capability which heavily depends on the application domain. Recent development in associative processing applications are also discussed which include fast routing in communication networks, memory management, database management, image processing, and artificial intelligence applications.

Jalaleddine-SMS; Johnson-LG , **Associative IC memories with relational search and nearest-match capabilities**, IEEE-Journal-of-Solid-State-Circuits. vol.27, no.6; June 1992; p.892-900.

The authors present the design and implementation of content addressable memories (CAMs) that execute relational and nearest-match instructions. Implementation of a novel relational search cell is presented. Direct performance comparison shows an order-of-magnitude improvement over existing designs with similar cell area. The design and implementation of a neural-inspired nearest-match CAM using a winner-take-all (WTA) network is presented. An original approach to analyzing such neural-inspired CAMs is presented. A model which describes the behavior of the WTA network is derived to be utilized in the design and performance prediction of the network. Performance of the WTA network in differentiating between words with large bit mismatches is analyzed, and an upper bound is set. Fully functional prototype chips have been fabricated through MOSIS using 2- mu m double-metal CMOS technology. Theoretical, simulation, and physical chip measurements are in good agreement.

Lea-RM , **Building blocks for associative memory**, Electronic-Engineering. vol.49, no.593; June 1977; p.77, 79-80.

Currently available content addressable memories are static designs using bipolar or CMOS processes. The author argues that dynamic MOS CAMS offer superior characteristics.

Schultz-KJ; Gulak-PG, **Architectures for large-capacity CAMs, Integration**, *The-VLSI-Journal,* vol.18, no.2-3; June 1995; p.151-71.

Content addressable memories (CAMs) have significantly lower capacities than RAMs. Following a summary of large-capacity CAM applications and a brief tutorial look at CAM operation, this paper reviews the sources of this capacity disadvantage: comparator area overhead and difficulty implementing two-dimensional decoding. Past attempts at achieving higher CAM density and capacity are reviewed, and advantages and disadvantages of each are

discussed qualitatively. Architectures are divided into the broad classes of serial and fully parallel. The former include bit-serial, orthogonal-RAM-based, insertion-memory, word-serial, multiport, vector-centered, pattern-addressable memory, and systolic associative memory. The latter include standard architectures, post-encoding, and pre-classification. A taxonomy, providing the first structured comparison of existing techniques, is presented. Thereafter, four architectures (two serial and two fully parallel) are quantitatively analyzed, in terms of delay, area, and power, and the cost-performance measures area x delay and power x delay. The fully-parallel architectures, despite their high cost, produce superior cost-performance results.

Optical Content Addressable Memory

Akyokus-S; Bruce-Berra-P , **A data/knowledge base machine based on an optical content addressable memory,** *Optical-Computing-&-Processing*, vol.2, no.3; July-Sept. 1992; p.179-87.

Abstract: Content addressing methods and associative architectures, with many different designs and structures, have been proposed for and used in database and knowledge base processing. This paper presents an optical content addressable memory data/knowledge base machine (OCAM-DM). The OCAM-DM uses a page-oriented optical mass memory (POHM) which provides a fast access rate and a large transfer rate. The authors expect that the proposed machine will provide at least two to three orders of magnitude performance improvement in the implementation of relational database operations over existing data/knowledge base machines assuming the availability of fast spatial light modulators and a POHM system.

Akyokus-S; Berra-PB, **Optical content addressable memories for data/knowledge base processing,** *Proceedings. The Fifth International Parallel Processing Symposium (Cat. No.91TH0363-2). IEEE Comput. Soc. Press, Los Alamitos, CA,* USA; 1991; xiv+656 pp. p.202-7.

Abstract: This paper describes the principles and architectures of a one- and a two-dimensional optical content addressable memory (OCAM). These architectures are based on optical matrix multipliers which use free space interconnections in optics. They work as a CAM due to spatial coding used to perform bit matching and masking. A 1-D CAM can perform a single (one-to-many) search operation and a 2-D OCAM can perform multiple (many-to-many) search operations in one step. An optical database machine (ODM) based on a 2-D OCAM along with a page-oriented holographic memory (POHM) is proposed for the execution of relational database operations.

Akyokus-S; Berra-PB, **A 3D optical database machine,** *Applications of Photonic Technology 2. Communications, Sensing, Materials, and Signal Processing* Plenum Press, New York, NY, USA; 1997; xvii+927 pp. p.543-51.

Abstract: This paper presents a 3D optical database machine that enables the parallel implementation of relational database operations. The basic data element in the proposed system is a 2D data page. The data pages are stored in a page-oriented optical mass memory, and processed by an 2D optical content addressable memory (2D OCAM) in parallel. A 2D OCAM enables the comparison of a 2D search page with a 2D data page in parallel. Given a search page and a data page of size n*m bits, a 2D OCAM call perform n^2m bit comparisons in single step (if n=512 and m=256, then n^2*m=671, 108, 864).

Awwal-AAS; Ahmed-JU, **Two-bit restricted signed-digit quaternary full adder,** *Proceedings of the IEEE 1994 National Aerospace and Electronics Conference NAECON 1994 (Cat. No.94CH3431-4). IEEE, New York, NY, USA;* 1994; 2 vol. xviii+1346 pp. p.1119-25 vol.2.

A multibit quaternary full adder is designed using a restricted set of modified signed-digit quaternary number system; Such adders could be stacked in parallel and they can add two n-bit numbers in a constant time irrespective of the operand size. An optical nonholographic content addressable memory is proposed for implementing the adder.

Belov-MN; Manykin-EA, **Principles of the design of an optical associative memory based on coherent four-waver mixing**, *Telecommunications-and-Radio-Engineering,-Part-2-(Radio-Engineering).* vol.48, no.4; April 1993; p.109-14 Translated from: Radiotekhnika. .

A new approach to the construction of associative memory systems in optics based on delayed four-wave mixing is proposed. The close analogy between the physical properties of this nonlinear optical medium and the features of learning and recognition in neural-net models with non-instantaneous interconnections is demonstrated.

Burtsev-VA; Fedorov-VB, **An optical associative memory for database management systems and computers with nontraditional architectures**
Telecommunications-and-Radio-Engineering,-Part-2-(Radio-Engineering). vol.47, no.7; July 1992; p.131-9 Translated from: Radiotekhnika. .

The problems of developing an optical associative memory for high-speed digital computers are examined, the limiting parameters of such a memory are estimated, and different design versions are analyzed.

Cherri-AK,**Symmetrically recoded modified signed-digit optical addition and subtraction**, *Applied-Optics.* vol.33, no.20; 10 July 1994; p.4378-82.

Symmetrically recoded modified signed-digit number algorithms are proposed for carry-free arithmetic. The superiority of the new algorithms is established over all other two-step-based symbolic-substitution optical implementations.

Javidi-B, **Programmable optical associative memory,** *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering.* vol.960; 1989; p.233-41.

A programmable optical associative memory for two-dimensional image retrieval is described. Both the stored image and the input image are displayed spatially; therefore, they can be updated in real time more conveniently. The integral product between the input image and the stored images is obtained by a nonlinear correlation technique which has a superior performance compared with the conventional optical correlation techniques in the areas of the light efficiency and the correlation signal quality. Thus, better quality images can be reconstructed and the need for optical gain and the optical feedback may be eliminated.

Johnson-KM; Kranzdorf-M; Bigner-BJ; Zhang-L , **Optical associative memory utilizing electrically and optically addressed liquid crystal spatial light modulators**, *Optical Computing 1989 Technical Digest Series, Vol.9. Postconference Edition. Summaries of Papers Presented at the Optical Computing Topical Meeting*. Opt. Soc. America, Washington, DC, USA; 1989; xi+446 pp. p.32-5.

The goal of this research is to build optical connectionist machines (OCMs) capable of interconnecting many input units to output units via two-dimensional liquid crystal spatial light modulators (SLMs). Liquid crystals SLMs are chosen because of their low optical absorption and power dissipation, moderate switching speeds, potential for high extinction ratios and resolution. In addition, these materials are birefringent and can easily implement a polarization-based optical associative memory. Previously, the authors presented results on connecting five input units to five output units, and eight input units to eight output units with the OCM. The number of interconnections was limited by the low extinction ratio of the two-dimensional SLM used as the OCM connection matrix (The Radio Shack Pocketvision Model 5 extinction is 5:1). In this paper they present results of using higher contrast computer controlled SEIKO liquid crystal pocket television model LVD.012 and optically addressable ferroelectric liquid crystal SLMs to hetero-associate pairs of 32 bit long input and output vectors. This association is performed using the least mean square algorithm (LMS), implemented with polarization encoding to represent both positive and negative weights.

Kosko-B; Guest-C, **Optical bidirectional associative memories,** Proceedings-of-the-SPIE – The-International-Society-for-Optical-Engineering. vol.758; 1987; p.11-18.

Four optical implementations of bidirectional **associative** memories (BAMs) are presented. BAMs are heteroassociative **content addressable memories** (CAMs). A BAM stores the m binary associations $(A_1, B_1), \ldots, (A_m, B_m)$, where A, B are points in the Boolean n-cube, p-cube respectively. A, B are neural networks of n, p bivalent or continuous neurons. The fixed synaptic connections between the A and B networks are represented by an n-by-p real matrix M. Bidirectionality of information flow in neural nets produces two-way **associative** search for the nearest stored pair to an input key. When the BAM neurons are activated, the network quickly evolves to a stable state of two-pattern reverberation, or pseudo-adaptive resonance. The BAM storage capacity for reliable recall is roughly m<min (n, p)-pattern number is bounded by pattern dimensionality. BAM optical implementations are divided into two approaches: matrix vector multipliers and holographic correlators. The four optical BAMs described respectively emphasize a spatial light modulator, laser diodes and high-speed detectors, a reflection hologram, and a transmission hologram.

Louri-A, **Design of an optical content-addressable parallel processor with applications to fast searching and information retrieval**, *Proceedings. The Fifth International Parallel Processing Symposium (Cat. No.91TH0363-2).* IEEE Comput. Soc. Press, Los Alamitos, CA, USA; 1991; xiv+656 pp. p.234-9.

Associative processing based on content-addressable memories has been argued to be the natural

solution for non-numerical information processing applications. Unfortunately, the implementation requirements of these architectures using conventional electronic technology have been very cost prohibitive, and therefore associative processors have not been realized. Optics has the capability over electronics for directly supporting associative processing by providing economic and efficient interconnects, massive parallelism, and high-speed processing. This paper presents the principles of designing an optical content-addressable parallel processor called OCAPP, for the efficient support of parallel symbolic computing. The architecture is designed to fully exploit optics advantages in interconnects and high-speed operations, and is potentially very suitable for applications where the number of data sets to be operated on is high. Several search and retrieval algorithms are mapped onto OCAPP to illustrate its ability to support parallel symbolic computing.

Murdocca-M; Hall-J; Levy-S; Smith-D, **Proposal for an optical content addressable memory**, *Optical Computing 1989 Technical Digest Series, Vol.9. Postconference edition. Summaries of Papers Presented at the Optical Computing Topical Meeting. Opt. Soc. America, Washington, DC, USA*; 1989; xi+446 pp. p.210-13.

A content addressable memory (CAM) design that demands high throughput is proposed for arrays of optically nonlinear logic gates interconnected in free space, with simple components such as spherical lenses, mirrors, and gratings. The design is suitable for an optical implementation because the bandwidth requirements cannot be met with electronics technology in the foreseeable future, and because the optical interconnection scheme presented is simpler than competing optical interconnect topologies such as fibers and one-to-many schemes based on holograms or magnification.

Oita-M; Nitta-Y; Tai-S; Kyuma-K, **Optical associative memory using optoelectronic neurochips for image processing,** *IEICE-Transactions-on-Electronics*. vol.E77-C, no.1; Jan. 1994; p.56-62.

This paper presents a novel model of optical associative memory using an optoelectronic neurochips, which detects and processes a two-dimensional input image at the same time. The original point of this model is that the optoelectronic neurochips allow direct image processing in terms of parallel input/output interface and parallel neural processing. The operation principle is based on the nonlinear transformation of the input image to the corresponding the point attractor of a fully connected neural network. The learning algorithm is the simulated annealing and the energy of the network state is used as its cost function. The computer simulations show its usefulness and that the maximum number of stored images is 150 in the network with 64 neurons. Moreover, we experimentally demonstrate an optical implementation of the model using the optoelectronic neurochip. The chip consists of two-dimensional array of variable sensitivity photodetectors with 8*16 elements. The experimental results shows that 3 images of size 8*8 were successfully stored in the system. In the case of the input image of size 64*64, the estimated processing speed is 100 times higher than that of the conventional optoelectronic neurochips.

Scarr,-R.W, **Associative memory systems** Oct 20, 1987.US-4701879

An associative optical memory system comprising a matched optical holographic filter, including an input plane comprised by a liquid crystal over silicon display, a Fourier transform plane and an output plane, the planes being separated by thin spherical lenses, and a coherent light source, and a digital computing system including a processor and a direct memory access element loaded by a backup store, which computing system is coupled to the filter for controlling the input and Fourier transform planes and the light source for parallel optical processing of the memory content, the output plane providing information to the processor as to the location of occurrences in the memory of a searched for item, wherein in use of the memory system a page of the memory content is loaded into the input plane, stored and the display blanked, a representation of the item to be recognized is loaded into a defined area of the display and a hologram thereof recorded in the Fourier transform plane by a pulsed reference beam derived from the coherent light source, wherein the loaded page is then displayed and illuminated by another beam derived from the coherent light source, wherein the light is filtered in the Fourier transform plane, detected at the output plane, decoded and passed to the processor for use in determining the location in the memory of the input display of the occurrences.

Holographic Associative Memory:

Akahori-H, **Information storage and search using holography**, *Researches-of-the-Electrotechnical-Laboratory*. no.771; Oct. 1977; p.1-100.

A survey of researches on holographic content-addressable memory is presented. A model of the holographic information search system based on detailed investigations of the system organisation is presented and the whole aspect of the method is explained. Methods of enhancing the performance of the holographic information search system by improving the diffraction efficiency of the information search hologram are dealt with. Analysis concerning phase coding is also presented concluding that the deterministic phase coding is much more advantageous than random phase coding. As an example, the construction of a dictionary having capability of high speed interrogation is proposed.

Azgomi-S, **Content addressable memory (CAM) and its applications**, *Electronic-Engineering*, vol.71, no.871; Aug. 1999; p.23-4, 26, 28.

Abstract: For communications system designers, using and understanding content addressable memory CAM, has taken on a new importance. This is because CAM, is ideally suited for many functions, including Ethernet address look-up, data compression, pattern-recognition, cache tags, high-bandwidth address filtering, and fast look-up of routing, user privilege, security or encryption information on a packet-by-packet basis for high-performance data switches, firewalls, bridges and routers. This article discusses several of these applications as well as hardware options for using CAM.

Caulfield-HJ, **Associative mappings by optical holography,** *Optics-Communications*, vol.55, no.2; 15 Aug. 1985; p.80-2.

Algebra-based content addressable memories have been constructed recently using optical methods. The author shows that holography can accomplish the same tasks on much larger vectors. The primary advantage is the size of vector or image, which can be handled. Other advantages include the fact that no calculations are required, it is easy to update or rewrite and can carry out 'instant' calculations.

Changsuk-Oh; **Hankyu-Park, Real-time Fourier transformed holographic associative memory with photorefractive material**, *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering*. vol.963; 1989; p.554-9.

The authors describe a volume holographic associative memory using photorefractive material and conventional planar mirror. The multiple hologram is generated with two angular multiplexed writing beams and a Fourier transformed object beam in $BaTiO_3$ crystals at 0.6328 mu m. The complete image can be recalled successfully by partial input of the originally stored image. It is proved that the system is useful for the optical implementation of real-time associative memory and location addressable memory.

Chen-Z; Kasamatsu-T; Shimizu-M; Shiosaki-T, **Optical associative memory using photorefractive $LiNbO_3$:Fe hologram, and $LiNbO_3$:Fe and $BaTiO_3$ phase onjugate mirror,** *ISAF '92. Proceedings of the Eighth IEEE International Symposium on Applications of Ferroelectrics (Cat. No.92CH3080-9),* IEEE, New York, NY, USA; 1992; xii+644 pp. p.47-50.

Experimental results are presented for phase conjugate (PC) wave generation, and the possibility of an application to an optical associative memory is discussed. Using an iron-doped lithium niobate single crystal, greater than 300% reflection efficiency was achieved by the degenerate four-wave mixing method, which was almost independent of the total incident optical intensity. The generator was then applied as a PC mirror to an all-optical associative holographic memory to reflect and amplify the diffracted beams. The retrieved output image was formed by using partial input illumination. Using a barium titanate single crystal, a PC reflection efficiency of about 70% was obtained by the self-pumping method. In order to accelerate the speed of the generation, a novel method called accelerated self-pumped phase conjugation (SPPC) was used. It was demonstrated that the accelerated SPPC could be directly applied to an optical associative memory to reflect the diffracted beams if a readout beam with a power of 40 mu W was used.

Cherri-AK; Karim-MA, **Symbolic-substitution-based operations using holograms: multiplication and histogram equalization**, *Optical-Engineering*. vol.28, no.6; June 1989; p.638-42.

Holographic content-addressable memory is used in an optical symbolic-substitution-based system to realize an efficient and high speed optical multiplier. Such a system is also capable of

performing image processing operations. Consequently, it is used to obtain a fast gray-level histogram and histogram equalization of an input image.

Eichmann-G; Kostrzewski-A; Dai-Hyun-Kim; Yao-Li, **An optical-holographic-associative-memory-based parallel register transfer processor,** *Optical Computing 1989 Technical Digest Series*, Vol.9. Postconference Edition. Summaries of Papers Presented at the Optical Computing Topical Meeting. Opt. Soc. America, Washington, DC, USA; 1989; xi+446 pp. p.240-3.

A specific hybrid sequential computing module, where optical array processors that perform the combinatorial logic and interconnect operations, are sandwiched between high-speed electronic parallely-addressed storage registers, is described. This hybrid system can sustain various fast optical register transfer micro-operations (ORTMOs), operations that are the most primitive operations required for an optical digital computer. This new system will be referred to as an optical register transfer processor (ORTP).

Eun-Soo-Kim; Seung-Hyun-Lee; Woo-Sang-Lee, **Optical implementation of a holographic heteroassociative memory system,** *Japanese-Journal-of-Applied-Physics,-Part-2-(Letters),* vol.29, no.7; July 1990; p.1304-6.

A holographic heteroassociation system that can store and reconstruct a sequence of associative information, based on the optical associative memory loop is described. In this system, the associative information is stored in two multiplexed Fourier transform holograms with different sequential arrangements, and the reference beam forms an associative link between them. Depending on the spatial arrangement style, many interesting applications can be possible. Some experimental results on cyclic search using this system are demonstrated.

Eung-Gi-Paek; Demetri-Psaltis, **Optical associative memory using Fourier transform holograms,** *Optical-Engineering,* vol.26, no.5; May 1987; p.428-33.

An experimental demonstration of a holographic associative memory is presented. The system utilizes an array of classic VanderLugt correlators to implement in parallel the inner product between an input and a set of stored reference images. Each inner product is used to read out an associated image. Theoretical analysis of the system is given, and experimental results are shown.

Knight-G, **Holographic associative memory and processor**, *Applied-Optics.* vol.14, no.5; May 1975; p.1088-92.

Describes a format of holographic associative memory that uses destructive interference to provide a null signal output for an exact match. For simplicity, the analysis given is for a one-dimensional page composer. A technique is given for parallel digital data processing with the memory described. The readout signal in the new format is not affected by page size.

Knight-GR, **Page-oriented associative holographic memory**, *Applied-Optics*. vol.13, no.4; April 1974; p.904-12.

A two dimensional page storage system is described which can perform many types of parallel searches on a large data base. The data can be located with one flash of a laser and a selected page can be read out on a second flash. Mathematical analyses of a page-oriented associative memory and an optical correlation memory are given. Problems associated with both types are discussed.

Paek,-E.G.; Psaltis,-D, **Optical associative memory using Fourier transform holograms,** 8 pp., May 1987, Pub. No: AD-A184 630/2/HCW.

An experimental demonstration of a holographic associative memory is presented. The system utilizes an array of class VanderLugt correlators to implement in parallel the inner product between an input and a set of stored reference images. Each inner product is used to read out an associated image. Theoretical analysis of the system is given, and experimental results are shown.

Soffer-BH; Marom-E; Owechko-Y; Dunning-G, **Holographic associative memory employing phase conjugation,** *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering,* vol.684; 1986; p.2-6.

The principle of information retrieval by association has been suggested as a basis for parallel computing and as the process by which human memory functions. Various associative processors have been proposed that use electronic or optical means. Optical schemes, in particular, those based on holographic principles, are well suited to associative processing because of their high parallelism and information throughout. Previous workers demonstrated that holographically stored images can be recalled by using relatively complicated reference images but did not utilize nonlinear feedback to reduce the large cross talk that results when multiple objects are stored and a partial or distorted input is used for retrieval. These earlier approaches were limited in their ability to reconstruct the output object faithfully from a partial input. The authors combine the principles of holographic memories and phase conjugation to implement a novel, all-optical holographic associative memory and symbolic processor.

Shifu-Yuan; Guofan-Jin; Minxian-Wu; Yingbai-Yan; Jingwen-Zhang; Kebin-Xu; Lixue-Chen, **Holographic associative memory with accurate addressing**, *Optical-Engineering*. vol.34, no.7; July 1995; p.2115-19.

We demonstrate the weakness of the real-time holographic associative memory implementation in addressing accuracy and describe a novel method to implement associative memory with

accurate addressing. In this method, the memory images and their background images are stored in two common holographic memory systems, and the partial image and its background image are simultaneously used to address the two holographic memory systems. The experimental results have proved the system feasible. In addition, we describe a method of associative memory with accurate addressing in a common real-time holographic memory system using combinatorial memory images and combinatorial addressing images.

Yariv-A; Sze-Keung-Kwong; Kyuma-K, **Demonstration of an all-optical associative holographic memory**, *Applied-Physics-Letters*. vol.48, no.17; 28 April 1986; p.1114-16.

The authors describe a new type of associative holographic memory and some related results. The memory can retrieve overlapping images which are stored in a volume hologram. The storage and retrieval of more than one image is demonstrated for the first time. The important role of thresholding in assuring 'clean' retrieval is demonstrated.

Yi-Mo-Zhang; Wei-Liu; He-Qiao-Li, **Holographic associative memory with parallel processing architecture**, *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering*. vol.2849; 1996; p.271-9

Abstract: In this paper, a multi-channel associative memory instrumentation prototype was presented, which was divided into two parts spatially. The lower part is a one-channel system using diode-pumped solid-state laser for demonstrating the holographic associative memory miniaturization, while the upper one is a four-channel system for proving the parallel processing architecture. Each channel is actually a photorefractive-based associative memory with external storage and using liquid crystal switch as threshold device in correlation domain.

Two Photon Memory

Esener-S; Fainman-Y; Ford-J; Hunter-S, **Two photon three dimensional memory hierarchy**, *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering,* vol.1773; 1993; p.346-55.

The computational power of current high-performance computers is increasingly limited by data storage and recall rates. In existing sequential-access electronic memories, a hierarchy of devices from cache memory to secondary storage provides performance continuum, allowing a balanced system design. The authors discuss the use of 3-D volume storage based on two photon materials to bridge the gaps in the storage hierarchy of parallel-access memories.

Hunter-S; Kiamilev-F; Esener-S, **Two-photon three-dimensional memory**, *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering,* vol.1291; 1990; p.113-18.

A new type of memory device is presented which takes advantage of the volume of a storage material in order to achieve extremely high information density and capacity. The unique properties of two-photon materials allows for reading and writing to any localized region

throughout the volume of material. In addition to the high capacity, the 2-photon 3-D memory system has been designed to access up to 10^6 bits in a single clock cycle. This large parallelism, combined with an access time of 1 mu sec, gives a memory bandwidth of 10^{12} bits/sec. It is shown that this value of memory bandwidth far exceeds that available from current memory systems and therefore is well-suited for the demands of current and future supercomputing systems.

Piyaket-R; Cokgor-I; Esener-SC; Solomon-C; Hunter-S; Ford-JE; Dvornikov-AS; Tomov-I; Rentzepis-PM, **Three-dimensional memory system based on two-photon absorption**, *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering,* vol.2297; 1994; p.435-46.

We discuss a volume optical storage 3D memory based on a two-photon absorption process. This memory has an advantage over a 2D memory in that it combines fast access time with high memory capacity, a capability that 2D memories cannot deliver. The two-photon 3D memory allows data to be accessed in parallel, and thus speeds up data transfer rate significantly. In this paper, we address the issue of memory hierarchy which has been organized to provide a performance continuum of different memory technologies available at present. We justify the two-photon 3D memory as a viable technology to fill the performance gap which currently exists between primary and secondary storage systems. We also discuss system capacity pertaining to two unique addressing schemes, i.e. orthogonal beam addressing and counter-propagating beam addressing. Finally, we report progress in system study regarding the effect of memory material characteristics on system design and development. Factors such as optimum wavelengths for read/write operations, material fluorescence, material fatigue, and the concentration of active molecules in the host material were considered.

Optical Disk based Associative Memory

Krishnamoorthy-AV; Marchand-PJ; Yayla-G; Esener-SC, **Photonic content-addressable memory system that uses a parallel-readout optical disk**, *Applied-Optics.* vol.34, no.32; 10 Nov. 1995; p.7621-38.

We describe a high-performance associative-memory system that can be implemented by means of an optical disk modified for parallel readout and a custom-designed silicon integrated circuit with parallel optical input. The system can achieve associative recall on 128*128 bit images and also on variable-size subimages. The system's behavior and performance are evaluated on the basis of experimental results on a motionless-head parallel-readout optical-disk system, logic simulations of the very-large-scale integrated chip, and a software emulation of the overall system.

Marchand-PJ; Ambs-P, **Developing a parallel-readout optical-disk system,** *IEEE-Micro.* vol.14, no.6; Dec. 1994; p.20-7.

Optical-disk technology offers high storage densities with the possibility of parallel readout. Simply by illuminating a disk area containing more than one bit and appropriately encoding the bits, this technology permits retrieval of multiple data in parallel. When used in conjunction with

a custom-designed parallel optoelectronic processing system, such systems meet the front-end storage and processing requirements for parallel-computing applications. The content-addressable memory system we are developing should prove useful for future very large database systems.

Marchand-PJ; Krishnamoorthy-AV; Urquhart-S; Ambs-P; Esener-SC; Lee-SH, **Motionless-head parallel readout optical-disk system,** *Applied-Optics.* vol.32, no.2; 10 Jan. 1993; p.190-203.

The design, analysis, and feasibility of a novel motionless-head parallel readout optical-disk system are presented. The system is designed to read data blocks distributed radially on the disk's active surface, and it has the unique advantage that no mechanical motion of the head is required for fast access, focusing, or tracking. Data access is achieved solely through the disk rotation, and the entire memory can be read in one rotation. In principle, this permits a data rate of up to 1 Gbyte/s. The data blocks are one-dimensional Fourier-transform computer-generated holograms, each reconstructing one column of a two-dimensional output image. Owing to the information redundancy and shift invariance properties of Fourier-transform holograms, tracking and focusing servo requirements are eliminated. A holographic encoding method is developed to produce high signal-to-noise ratio reconstructions and to reduce significantly the radial alignment requirements of the recorded data bits. The optical readout system consists of only three cylindrical lenses. Two of these may be replaced by a single hybrid diffractive-refractive optical element for easier system alignment and better optical performance, i.e. reduced aberrations and improved resolution. The throughputs and retrieval times of this parallel readout optical-disk system make it well suited to a variety of parallel computing architectures, including a high-performance optoelectronic associative memory.

Taiwei-Lu; Kyusun-Choi; Shudong-Wu; Xin-Xu; Yu-FTS, **Optical disk based neural network,** *Applied-Optics.* vol.28, no.22; 15 Nov. 1989; p.4722-4.
Using an optical disk as a large capacity associative memory in an optical neural network is described. The proposed architecture is capable of data processing at high speed.

Non-Holographic Content Addressable Memory:

Ahmed-JU; Awwal-AAS; Karim-MA, **Two-bit trinary full adder design based on restricted signed-digit numbers**, *Optics-and-Laser-Technology,* vol.26, no.4; Aug. 1994; p.225-8.

A 2-bit trinary full adder using a restricted set of a modified signed-digit trinary numeric system is designed. When cascaded together to design a multi-bit adder machine, the resulting system is able to operate at a speed independent of the size of the operands. An optical non-holographic content addressable memory based on binary coded arithmetic is considered for implementing the proposed adder.

Kostrzewski-A; Eichmann-G; Dai-Hyun-Kim; Yao-Li, **Parallel optical content addressable memory (CAM) arithmetic multiprocessor**, *International-Journal-of-Optical-Computing,* vol.1, no.1; Oct. 1990; p.5-24.

A new scheme for digital optical computing, utilizing a non-holographic optoelectronic content-addressable memory (CAM), is discussed. To illustrate the application of this arithmetic processor, the design of an optical binary carry look-ahead adder (CLA), and also, the design of a binary number (BN), a sign/logarithm number (SLN) and a residue number (RN) multipliers are presented. Compared to other existing approaches, this optoelectronic CAM offers a number of practical advantages, such as fast processing speed, ease of optical implementation and alignment. Active low and high spatial CAM mask encoding techniques are discussed. A multioperation and multibit CAM processor is described. Experimental results for a CLA and a BN, RN and SLN multipliers are presented.

Kostrzewski-A; Yao-Li; Eichmann-G; Dai-Hyun-Kim, **Fast optical digital arithmetic processors**, *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering.* vol.1296; 1990; p.332-43.

A new scheme for digital optical computing, utilizing a non-holographic opto-electronic content addressable memory (CAM), is discussed. to illustrate the performance of this arithmetic processor, the design of an optical binary carry look-ahead adder (CLA), also, the design of a binary, a logarithmic number (LN) and a residue number (RN) multiplier are presented. Compared to existing opto-electronic approaches, this non-holographic CAM offers a number of practical advantages, such as fast processing speed, ease of optical implementation and alignment. Two spatial input data encoding techniques, an active low and high, are discussed. A multioperation multibit CAM processor is presented. Experimental results for a CLA adders; and a binary, residue and logarithmic number multipliers are also presented.

Yao-Li; Dai-Hyun-Kim; Kostrzewski-A; Eichmann-G**, Content-addressable-memory-based single-stage optical modified-signed-digit arithmetic,** *Optics-Letters,* vol.14, no.22; 15 Nov. 1989; p.1254-6.

Using a novel nonholographic optoelectronic content-addressable memory in a free-space angular multiplexing geometry, a single-optical-stage compact parallel optical modified-signed-digit arithmetic processing architecture is proposed. Some spatial light modulator based experimental results are also presented.

Yao-Li; Dai-Hyun-Kim; Kostrzewski-A; Eichmann-G Hybrid**, content addressable memory MSD arithmetic**, *Proceedings-of-the-SPIE --The-International-Society-for-Optical-Engineering,* vol.1215; 1990; p.305-12.

The modified signed-digit (MSD) number system, because of its inherent weak interdigit dependence, has been suggested as a useful means for a fast and parallel digital arithmetic. To maintain a fast processing speed, a single-stage holographic optical content-addressable memory

(CAM) based MSD algorithm was suggested. A novel non-holographic opto-electronic CAM based fast MSD addition processing architecture is proposed. The proposed concept has been verified with the authors' first-order proof-of-principle experiments. A figure of merit comparison of this and other existing approaches is also presented. Based on this key opto-electronic CAM element, implementation of more sophisticated MSD arithmetic, such as optical MSD subtraction and multiplication operations are proposed.

Others

Mitkas-PA; Irakliotis-LJ; Beyette-FR-Jr; Feld-SA; Wilmsen-CW, **Optoelectronic look-up table using VCSEL-based logic,** *LEOS '93 Conference Proceedings. IEEE Lasers and Electro-Optics Society 1993 Annual Meeting (Cat. No.93CH3297-9).* IEEE, New York, NY, USA; 1993; xxvi+832 pp. p.71-2.

Look-up tables, also known as truth tables, have been commonly used in several processing tasks, such as database operations, **associative** processing, residue arithmetic, cache memories, mathematical function modules, and instruction decoding. In its simplest form, a look-up table contains a list of values that must be compared against an input argument. If the input value is found among the table entries, a match signal is generated. Many variations of this scheme exist, such as input/output tabulation, multiple and/or partial matching capability, **content addressable memories**, and auto- and heteroassociative memories. The main function of searching the look-up table is a highly parallel process and must be completed, preferably, in a single step. The low computational requirements of the look-up operation, coupled with its large degree of parallelism and tabular representation of data, have led to several implementations of optical look-up table architectures. In this paper, we describe an optoelectronic look-up table configuration based on an array of exclusive-or gates implemented with heterostructure phototransistors (HPT) and vertical cavity surface emitting lasers (VCSEL).

Selected References

[IMAGE1] F.P. Herman and C.G. Sodini, **A Dynamic Associative Processor for Machine Vision Applications**, IEEE Micro, Vol. 12 No. 3, pp31-41, June 1992.

[IMAGE2] Y. Shain, A. Akerib, R. Adar, **Associative Architecture for Fast DCT**, Proceedings of 1998 IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 5, pp 3109-3112, 1998.

[IMAGE3] W.E. Snyder and C.A. Savage, **Content-Addressable Read/Write Memories for Image Analysis**, IEEE Transactions on Computers, vol. C-31, No.10, pp 963-968, October 1982.

[IMAGE4] R. Storer, **Image Rendering with Content-Addressable Parallel Processors, in Associative Processors and Processing**, A. Krikelis and C. Weems, editors, IEEE Computer Society Press, Los Alamitos, California, 1997.

[DISK1] Psaltis D, Neifeld MA, Yamamura A, Kobayashi S, **Optical Memory Disks In Optical Information-Processing**, *Applied Optics*, Vol 29: (14) 2038-2057 MAY 10 1990

[DISK2] Mitkas-PA; Berra-PB, **PHOEBUS: an optoelectronic database machine based on parallel optical disks**, *Journal-of-Parallel-and-Distributed-Computing* vol. 17, no.3; March 1993; p.230-44.

[2PHO2] Pericles A. Mitkas and Leo J. Irakliotis, **Three-dimensional optical storage for database processing**, invited paper, *Optical Memory and Neural Networks*, Volume 3, Number 2, pp.217, 1994.

[2PHO3] S. Hunter, F. Kiamilev, S. Esener, D.A. Parthenopoulos, and P.M. Rentzepis, **Potientials of Two-photon Three-dimensional Optical Memories for High Performance Computting**, *Applied Optics*, Vol. 29, No. 14, pp2058-2066, May 1990.

[HOLO1] Demetri Pasaltis and Fai Mok, **Holographic Memories**, *Scientific American*, Vol23, No 55, pp. 70-76, Nov 1995.

[NONHOLO] Kostrzewski-A; Yao-Li; Dai-Hyun-Kim; Eichmann-G, **Non-holographic content addressable memory based arithmetic processor**, Proceedings-of-the-SPIE—The-International-Society-for-Optical-Engineering, vol. 1230; 1990; p.725-8.

*U.S. GOVERNMENT PRINTING OFFICE: 2002-710-038-10220

MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)

The advancement and application of Information Systems Science and Technology to meet Air Force unique requirements for Information Dominance and its transition to aerospace systems to meet Air Force needs.